Títol:
Monitorización de sistemas de producción

Volum: **1/1**
Alumne: **Ricardo Frías Álvarez**

Director/Ponent: **Pere Marés Martí**
Departament: **ESAII**
Data: **14 de diciembre del 2008**

Índice de contenido

1 Introducción

Actualmente en el mundo de la informática existen sistemas de monitorización que permiten de una forma económica, centralizada y fácil; supervisar el estado de servicios, máquinas y conexiones; desde cualquier parte del mundo. Los sistemas de monitorización se emplean para revisan de forma automática el estado de servicios y máquinas. En caso de que algún servicio o máquina no responde de forma apropiada, el sistema lo notifica mediante la vía definida según la gravedad de la situación (Notificaciones vía SMS, correo electrónico, llamada telefónica, etc).

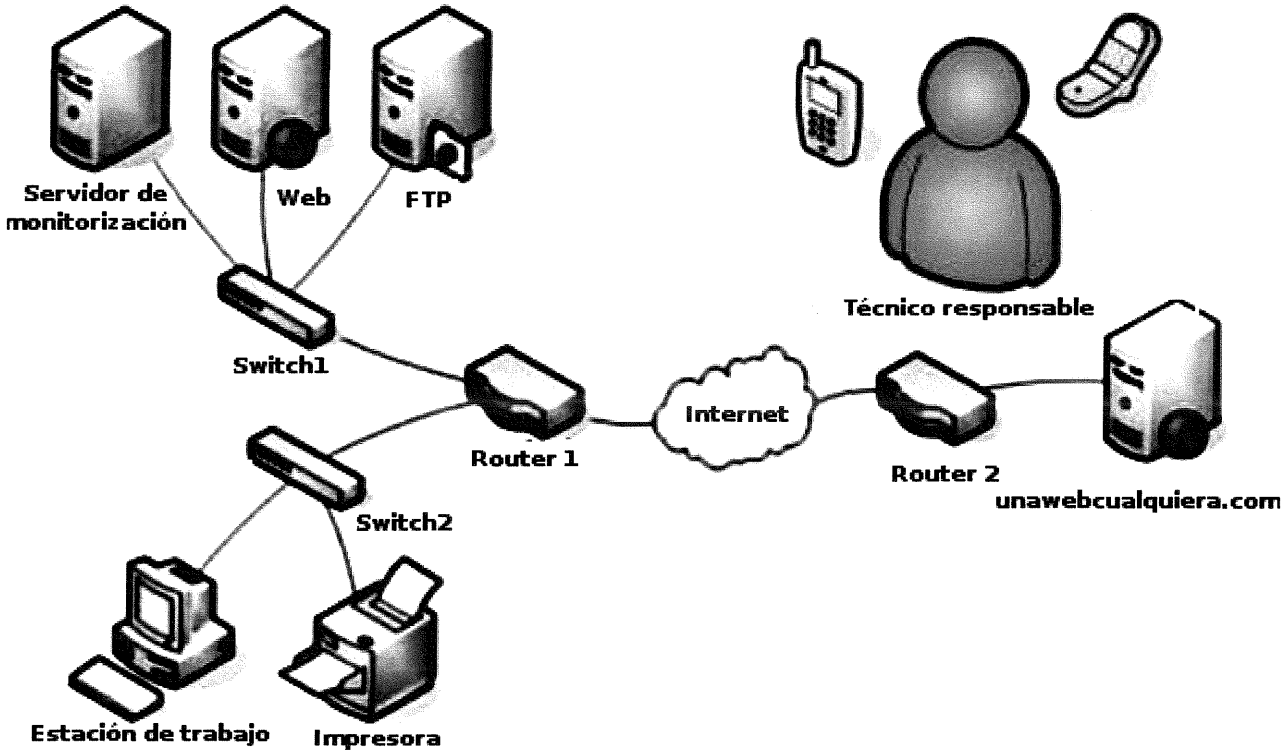

Ilustración 1: El servidor de monitorización se encarga de chequear los equipos e informar de cualquier anomalía localizada al técnico responsable.

Algunos de estos sistemas, además de enviar notificaciones cuando hay errores o se ha restablecido el servicio, disponen de una interfaz web que permite observar el estado de los diferentes servicios en cada momento.

La monitorización automática y centralizada está muy extendida en el mundo de la informática, sin embargo, no se da de igual medida en el sector de la producción en el cual es más común un sistema descentralizado (Hay que acceder a cada máquina para conocer su estado), método que no resulta tan económico, rápido y eficaz, como el sistema centralizado de monitorización. Uno de los principales inconvenientes de estos sistemas suele ser el coste de su implementación que requiere una inversión inicial importante.

Podemos ayudarnos de estos sistemas para gestionar las averías y reparaciones, en otras palabras actuar antes de que se produzcan averías. Muchas de las avería que sufren las máquinas las podríamos evitar si disponemos del sistema de monitorización que nos avise cuando los valores de los sensores sobre pasen su límite. Por ejemplo, si la capacidad del deposito de aceite de un motor es inferior al 5%, el sistema envía un SMS al técnico para que rellene el deposito.

Incluso con el debido software y los datos proporcionados por el sistema se puede llegar a obtener información muy útil y precisa sobre el rendimiento de producción, que puede ofrecer ventajas sobre los competidores (Piezas producidas/tiempo, paradas de las máquinas, motivo de la parada, tiempo en producción, etc).

Con este proyecto se trata de desarrollar un sistema genérico de monitorización centralizado, económico para las máquinas de producción, todo ello utilizando software libre y entorno linux.

1.1 Motivación

Desde antes de empezar la carrera, la interacción entre los equipos electrónicos y los PC es un tema que ha interesado. De hecho, mi trabajo de investigación de bachillerato se centró en este tema. Construí la maqueta de un puente levadizo que se elevaba o bajaba según el estado de dos sensores conectados a un PC. Un programa que corría sobre el PC se encargaba de decir cuando se tenía que elevar o bajar la plataforma.

También he de señalar que las dos asignaturas de la carrera que más me han interesado, están muy relacionadas con el presente proyecto, SDMI "Sistemas digitales y microcontroladores", PI "Periféricos e interficies".

En mis dos últimas experiencias laborales, trabajé con los sistemas de monitorización para monitorizar el estado de servidores. Una de mis labores dentro del equipo de la asignatura PXCSO "Proyecto de redes de computadoras y sistemas operativos", fue la instalación y configuración del sistema de monitorización Nagios.

La elección de la monitorización de máquinas está muy relacionada con mi primera experiencia laboral, fue en una fábrica de producción de cajas de cartón y envases. En ella, no existían sistemas de monitorización. Con lo que se daban situaciones peligrosas para las máquinas. Por ejemplo, cuando en una máquina se encallaba una caja, ésta seguía sin darse cuenta acumulando cajas que colisionaban y corría el peligro de autoaveriarse.

Otra motivación para llevar a cabo este proyecto es la aplicación de los conocimientos de casi todas las asignaturas de la carrera de sistemas, como son: conocimientos de redes, sistemas operativos, programación, sistemas digitales, etc.

1.2 Antecedentes

Hasta hace unos años el sector de la producción en España, tenía muy lejos la competencia, que se hallaba en países como China, Polonia, Rumania, etc. Los sistemas de producción existentes eran suficientes, gracias a la protección que nos ofrecía el mercado común de la UE. No obstante, la reciente incorporación a la UE de países del este como Polonia, Rumania; ha supuesto un duro golpe para el sector de la producción porque estos países no encuentran barreras para exportar sus productos dentro de la UE.

Estos países cuentan con una mano de obra más barata, unos costes de producción inferiores y precios más atractivos. Por lo que son una dura competencia para el sector de la producción en España. Es ahora cuando nos estamos viendo obligados a incorporar sistemas para optimizar la producción, como el que se propone en este proyecto.

En el mercado existen muchos sistemas de monitorización de sistemas informáticos, como son Nagios, Zabbix, Bigsister o Cacti. Cada vez están extendiendo más, ya que permiten actuar antes de que surjan los problemas. Hacen posible el control de grandes redes de máquinas en tiempo real.

Normalmente casi todas las máquinas de un fábrica tienen sistemas de monitorización en su propio terminal, cuando algo falla se puede consultar el estado de los sensores desde terminal para encontrar el problema.

Existen sistemas en la industria productos predefinidos de monitorización centralizado para el mantenimiento preventivo, como es el caso de VIBRA de la empresa Sinais. La mayoría de los sistemas de monitorización para medios de producción están desarrollados por empresas especializadas que presentan el proyecto y lo adaptan a las necesidades de cada cliente.

1.3 Objetivos

El objetivo principal de este proyecto es conseguir un sistema universal, centralizado, de bajo coste y mantenimiento que permita detectar anomalías en el funcionamiento de máquinas de producción e informe de ellos en el mínimo tiempo posible al operario responsable. Con este sistema se pretende conseguir un funcionamiento continuo de la cadena de producción, en otras palabras que se produzcan el mínimo de paradas posibles y se mantenga en todo momento informado al usuario del estado del sistema.

Se trata de un sistema universal porqué no se diseñará para monitorizar un grupo especifico de máquinas, por lo tanto, se podrá monitorizar cualquier máquina bajo previa adaptación. Centralizado porque un único punto conoce el estado de todo el sistema.

A continuación se detallarán los objetivos de forma desglosada que se pretender conseguir con este proyecto.

1.3.1 Monitorización de los sensores y el estado de las máquinas

El primer objetivo consiste en conseguir un mecanismo de comunicación entre el sistema de monitorización y el equipo que vamos a monitorizar, para obtener el estado de los sensores y contadores del dispositivo.

Será necesario procesar esta información y compararla con los valores límite predefinidos para que el sistema pueda interpretar si son correctos o en que medida son anómalos.

1.3.2 Comunicación de alertas al usuario

El sistema tiene que ser capaz de comunicarnos un fallo o anomalía en el sistema, en un tiempo inferior a 5 minutos.

La interfaz que nos muestre el estado de las máquinas ha de ser intuitiva y accesible desde cualquier parte del mundo y dispositivo, por tanto, ha de ser accesible desde internet. Ha de ser intuitiva para que rápidamente se pueda actuar y detectar donde está el problema.

El servició de monitorización tiene que ser capaz de alertar al usuario mediante envío de correos electrónicos y alertas a teléfonos móviles.

Hay que conseguir que el sistema sea muy flexible y permita por ejemplo, medir la gravedad de la incidencia, el tipo de incidencia y en función de esto, use un medio u otro para notificar la incidencia al usuario. Según el tipo de servicio afectado el sistema ha de ser capaz de

distinguir e informar únicamente a las personas correspondientes.

1.3.3 Red segura de acceso externo

Se ha de garantizar una red de acceso restringido y seguro. En caso de que no se encuentre disponible el acceso a la red de monitorización porque está caída su conexión, tiene que existir un sistema de alerta alternativo que nos informe.

Remotamente tenemos que tener acceso a toda la información, tal y como si estuviésemos en la propia red.

1.3.4 Visualización del entorno de trabajo remotamente

Con el fin de evitar desplazamientos innecesarios y controlar la zona en remoto, se añadirán cámaras de vídeo en los puntos que se crea conveniente. Las imágenes captadas por estas cámaras serán accesibles desde internet. Se pretende su incorporación al sistema de monitorización, el sistema ha de detectar movimientos en zonas de acceso restringido y notificarlo al usuario.

1.4 Descripción resumida del proyecto

Al finalizar el proyecto se presentará la implementación de una maqueta que contenga al menos un represente de todos los elementos que se encontrarían en la implementación real del proyecto para una fábrica. En la siguiente ilustración aparece el diseño de la maqueta y a continuación de ella, la descripción de los diferentes elementos:

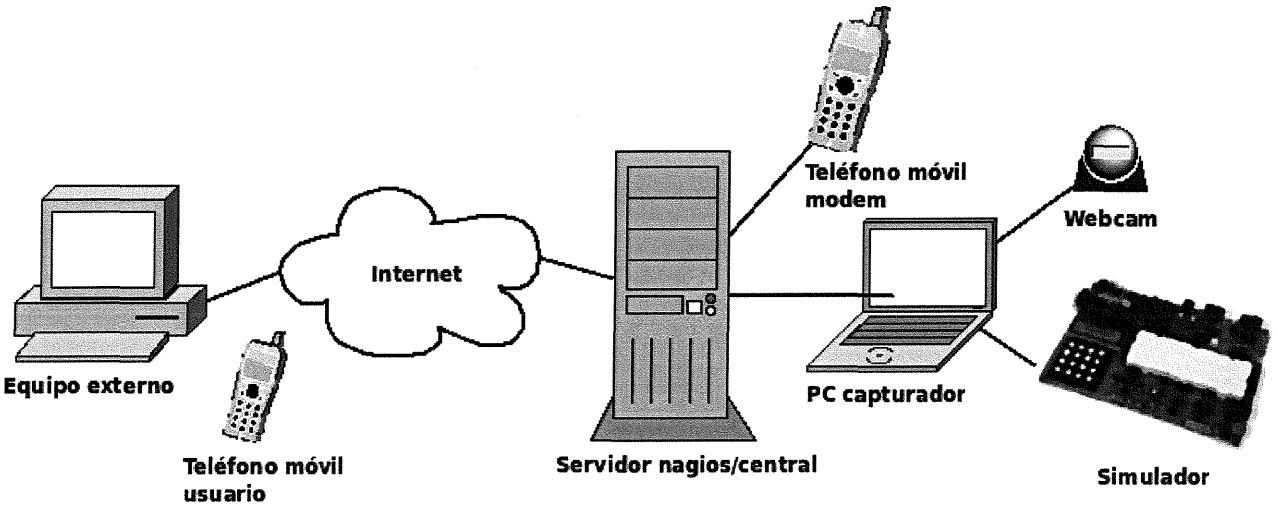

Ilustración 2: Esquema de la maqueta

- El equipo externo se encuentra conectado a la red de monitorización de forma segura mediante una VPN que permite al usuario comprobar el estado del entorno, acceder a la interfaz web del sistema de monitorización Nagios, descargar de correos de alertas, visualizar video y configurar las máquinas de la red.
- Un PC de sobremesa (Servidor Central) el cual tiene implementadas las funcionalidades de servidor central de monitorización con Nagios, servidor de correo local, servidor de conexión VPN, firewall de la red local.
- Un teléfono móvil conectado al servidor central que se emplea para enviar los avisos SMS y un teléfono móvil donde se reciben las alertas.
- El simulador se encarga de capturar el valor de los sensores y interruptores que tiene integrados. Además de responder a las peticiones del PC capturador.
- El PC capturador es el encargado de compartir por la red el valor de los sensores del simulador y de procesar las imágenes capturadas por la webcam.

El servidor Central consulta al PC capturador el valor de los sensores del simulador, si detecta un valor anormal, un equipo de la red caído o movimientos en la zona en un horario no permitido; envía un correo electrónico a la persona correspondiente y en función de la gravedad también envía un SMS a través del teléfono móvil.

Mediante los actuadores que dispone el simulador se puede variar el valor de los sensores y crear situaciones de alerta. Empleando los botones de navegación (Siguiente, anterior) del simulador se puede ver el valor devuelto por el sensor en tiempo real.

El siguiente esquema es un ejemplo de aplicación del diagrama de red de la maqueta y se asemejaría más a una aplicación real del sistema.

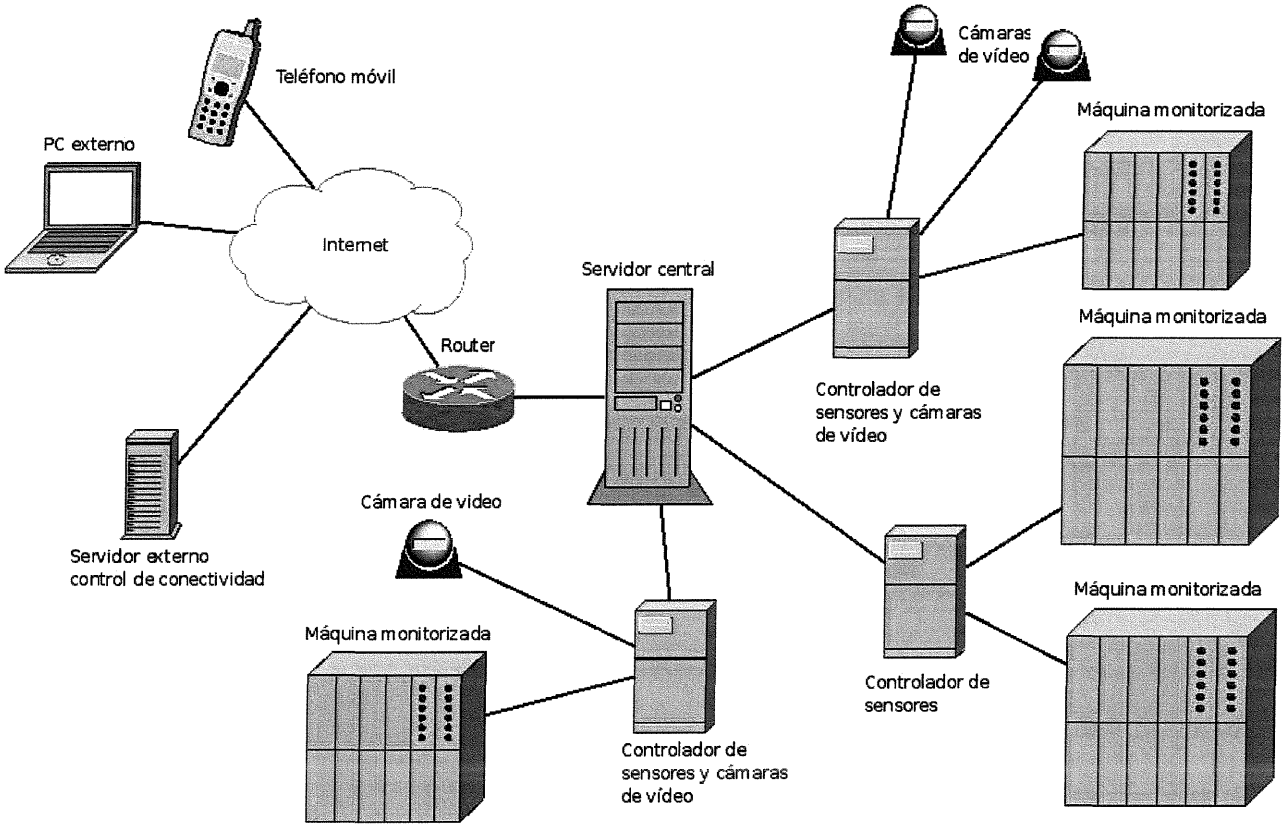

Ilustración 3: Esquema de la red de monitorización

2 Planificación del proyecto

A modo de demostración la implementación del sistema se llevará a cabo sobre una maqueta de pruebas que contendrá al menos un elemento de cada parte del sistema. Para simular los sensores y el equipo a monitorizar se empleará una placa de demostración.

Teniendo en cuenta los objetivos marcados, el proyecto ha de contemplar las etapas que se describen a continuación.

2.1 Etapas generales

2.1.1 Estudio de la viabilidad del proyecto

Antes de empezar a trabajar en el proyecto, es necesario conocer las diferentes etapas del proyecto para determinar si es viable. Es necesario tener en cuenta factores como el tiempo necesario para investigar, implementar las partes y el presupuesto. Asimismo se ha de valorar si empleando las tecnologías existentes, será posible cumplir los objetivos marcados.

Es necesario un estudio muy general de todas las partes del proyecto para familiarizarse con el entorno, los conceptos y poder realizar una planificación acertada.

2.1.2 Diseño de la estructura del sistema de monitorización

Una de los primeras funciones que se llevaran acabo será la elaboración del diseño de la estructura del sistema de monitorización. Esto permitirá conocer los elementos que serán necesarios para llevar acabo el proyecto y como están intercomunicados.

2.1.3 Estudio de la placa de demostración

Este punto consiste en conseguir una placa de demostración para simular los sensores y comunicarse con el PC. Necesitamos una placa que tenga al menos 4 sensores diferentes y permita la comunicación con el PC. Sería interesante que cuente con un interruptor o pulsador para simular la parada de emergencia. El rol de la placa dentro del esquema corresponderá con el de máquina monitorizada.

El siguiente paso, una vez escogida la placa, es familiarizarse con el entorno, para ello es preciso estudiar la arquitectura del microcontrolador incorporado en la placa y los componentes que componen la placa.

Ya conociendo completamente el entorno, hay que interpretar que función representará cada sensor, por ejemplo, el potenciómetro simulará el nivel de aceite de un deposito.

Finalmente se tendrán que analizar los métodos de comunicación entre PC y la placa de demostración, para poder implementarlos y transmitir el valor y estado de los sensores al PC.

2.1.4 Terminal local de captura máquina

Será preciso incorporar un terminal conectado directamente a la máquina (Placa de demostración), cuya función consistirá en recoger los datos de los sensores y compartirlos con Servidor central. Esta función está identificada en el esquema de red de la Ilustración 3 como "Controladores de sensores y servidor de vídeo".

Hay que escoger un sistema operativo ligero que permita la captura de vídeo, publicación de vídeo y compartir los datos de la placa. En la selección de las cámaras de vídeo se tendrá en cuenta el precio, que han de ser suficientes para tener una visión global de la zona, detectar movimientos en ella y han de ser compatibles con el sistema operativo instalado.

2.1.5 Instalación de servicios del servidor central

La función más importante que realizará el servidor central es la recogida de datos de los controladores de sensores, para poder cumplir esta función es necesario dotarlo de un sistema de monitorización en red muy flexible. Primero se estudiarán los diferentes sistemas de monitorización de redes existentes, para poder escoger el más apropiado. El equipo ha de ser capaz de comunicarse con el cliente empleando diferentes medios.

- Correo electrónico: El sistema a de ser capaz de enviar correos al cliente por lo que será necesario al menos instalar un agente de correo (MTA) que implante esta funcionalidad. Para escoger la mejor opción se analizaran las diferentes opciones disponibles para enviar correos.
- Mensajes cortos a teléfonos móviles (SMS): Consiste en el estudio de las diferentes opciones que hay disponibles para enviar SMS a teléfonos móviles, selección de la más conveniente y implementación.

- Interfaz accesible e intuitiva: Se ha de proporcionar una interfaz web al usuario para que pueda observar el estado de los diferentes servicios desde cualquier parte y dispositivo, normalmente la incorporan los sistemas de monitorización, por tanto se ha de analizar y decidir si es apropiada para nuestras necesidades, si es conveniente cambiarla, modificarla o diseñar una nueva.

El servidor central, también será el encargado de proporcionar una red segura de acceso restringido desde el exterior. Para ello, lo más acertado es el uso de una VPN, lo que implica la necesidad de investigar e implementar una VPN.

2.1.6 Servidor externo de control de conectividad

Hay que idear un servicio externo que en caso de caída de la línea principal, avise al usuario de que esta se ha caído, porque si se cae la conexión a internet no se recibirá ninguna alerta por correo y el sistema no será accesible.

2.1.7 Simulación y corrección de errores

Una vez acoplado todo el sistema llega el momento de probar que realmente funciona el sistema y corregir los errores que aparezcan.

Cuando todo el sistema funcione correctamente se preparará la demostración del proyecto, hay que tener en cuenta que en el menor tiempo posible se tiene que poder mostrar todas la funciones implementadas.

2.1.8 Documentación

La documentación del proyecto se irá escribiendo a medida de que este avance, ya que la idea es primeramente recolectar información, analizarla, documentar, implementar y por último, documentar de nuevo tras la implementación.

2.2 Diagrama de Gantt

A indicar que cada día del diagrama del Gantt equivale a 2 horas de trabajo más media hora de documentación. El proyecto se inicia el 24 de diciembre de 2007 y se prevé su finalización el 21 de mayo de 2008.

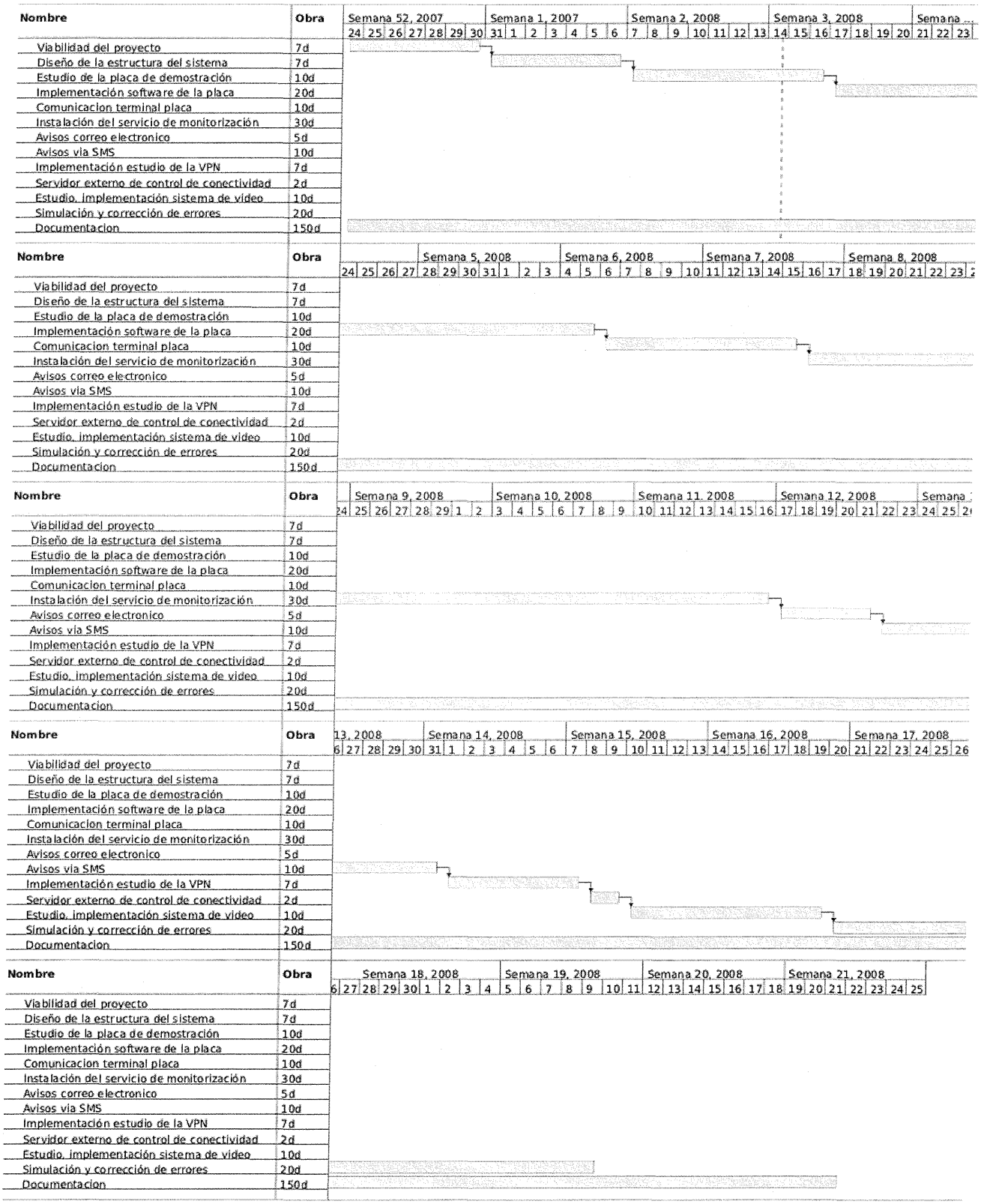

Ilustración 4: Diagrama de Gantt

3 Diseño del sistema de monitorización

Para que el diseño del sistema de monitorización sea el adecuado, hay que tener muy en cuenta todos objetivos marcados, para asegurarse que los recursos que se incluyen son los más ajustados y suficientes para cumplir todos los objetivos. La Ilustración 3 muestra el esquema del sistema de monitorización, es interesante tenerla presente durante este apartado.

Los primeros elementos del sistema, son las máquinas de producción, son los elementos que serán monitorizados, donde instalaremos los sensores y el medio por el cual obtendremos los datos. En el esquema de red aparecen como "Máquina monitorizada", están directamente conectadas a su controlador de sensores. En el modelo de demostración se sustituye la máquina de producción por un circuito programable (Nombrado simulador) que permite simular diversos sensores, analógicos, botones, pulsadores y secuenciadores lógicos. Por un precio asequible permite el ahorro del diseño de circuitos y facilita el trabajo con el microcontrolador.

El "Controlador de sensores y servidores de vídeo" es un PC, con un sistema operativo Linux. Dispone de diferentes interfaces que permiten la conexión con las máquinas que se han de controlar y las cámaras que controlará. Este equipo puede controlar una o más máquinas dentro del espacio de producción, en el caso de la demostración dispondremos de un PC para recoger, almacenar y compartir los datos del simulador con el "Servidor Central". Mediante un cable serie RS-232, y el protocolo UART recogeremos los datos del simulador porque es la única vía que nos ofrece el simulador. Esta máquina también se encarga de procesar y analizar el vídeo con el fin de detectar y grabar los movimientos en la zona. Para la maqueta de demostración conectaremos una única cámara USB.

El servidor central o "Servidor Central", es un ordenador con sistema operativo Linux, se encuentra conectado a los diferentes Controladores de sensores mediante red Ethernet (También existe la posibilidad de monitorizar máquinas situadas en otras delegaciones, saliendo a internet dentro de una VPN). Recoge toda la información que le facilitan los Controladores de sensores, la evalúa e informa al usuario en caso de ser necesario. También es el encargado de garantizar la seguridad dentro de la red, proporcionando un acceso restringido y seguro. Tanto en el modelo de demostración como en el real es un único equipo el que se encarga de todas estas tareas. El modelo centralizado nos ofrece mayores facilidades para su configuración, instalación y su posterior mantenimiento del sistema, ya que un sistema descentralizado de monitorización nos obligaría a instalar los servicios de monitorización en todas las máquinas.

Finalmente el servidor central puede estar conectado directamente a internet o puede conectarse a través de una Red Local(LAN). La conexión a internet nos permitirá, que los usuarios puedan consultar sus correos de alerta, acceso a la plataforma web y las cámaras, desde cualquier parte del mundo. También disponemos de un servidor externo de control de conectividad para confirmar que el servidor central se encuentra conectado a internet, de no ser así nos informaría.

El envío de SMS se puede realizar de diversas formas, mediante un Modem GSM o mediante servidores de internet.

4 Maqueta de simulación de una máquina de producción

4.1 Que es una placa de demostración

Una placa de demostración es un circuito electrónico compuesto por diferentes componentes que tienen como finalidad facilitar el aprendizaje del uso de los microcontroladores y/o facilitar el diseño y desarrollo de circuitos. En la imagen se puede observar la placa que se ha escogido para simular la máquina de producción en este proyecto.

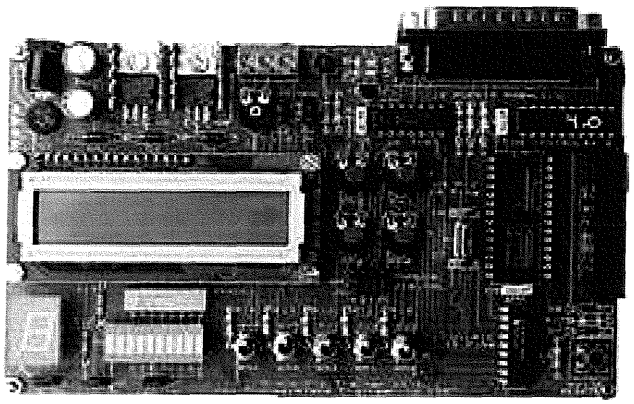

Ilustración 5: Placa de demostración

4.2 El porqué de la placa de demostración

Con el fin de conseguir un sistema transportable, económico y parecido a una máquina, se ha decidido que utilizar una placa de demostración es la opción que cumple en mayor medida estos objetivos.

Se podrían haber escogido otras opciones, como el uso de un PC que simulara los sensores pero se descarto esta posibilidad por los siguientes motivos:

- Los sensores también serían simulados, por lo que se ajustaría muy poco a la realidad.
- Es más voluminoso que una placa de demostración.
- Un PC portatil, tiene un precio más elevado que una placa de demostración.
- No es posible ni se dispone de otro PC.

Otra opción considerada fue el diseño e implementación de un circuito, pero también se descarto, por los siguientes inconvenientes:

- Demasiado costoso en tiempo.
- No entra dentro del temario de la carrera, por lo que requiere dedicar mucho tiempo para adquirir estos conocimientos.

4.3 Selección de la placa de demostración

El uso de la placa de demostración es por tanto la opción más viable. Pero para que sea realmente útil, ha de cumplir unos requisitos mínimos que se han tenido en cuenta en su selección.

Las placas de demostración suelen estar preparadas para trabajar con un fabricante, familia o microcontrolador específico. Existen múltiples fabricantes de microcontroladores, entre ellos los PIC de Microchip, Intel, Motorola, Zilog, etc. De entre ellos, los PIC de Microchip son los más usados para la formación, se trata una arquitectura ya conocida (Por lo que no será necesario aprender una nueva) y es fácil conseguir placas de desarrollo. Por estos motivos, todas las placas analizadas son para microcontroladores PIC.

A continuación se detallan los principales candidatos valorados y los motivos de la selección final del PIC'SCHOOL.

4.3.1 PICDEM.net 2 Development Board

Se trata de una una placa de desarrollo de la empresa Microchip, que permite conexiones TCP/IP mediante sus dos interfaces Ethernet. Además cuenta con un procesador de la gama alta de PIC el PIC18F97J60.

Las más interesantes funcionalidades que nos ofrece son: Sevidor Web compatible con HTTP, conexiones TCP/IP, 2 interfaces (RJ-45), pantalla LCD, botones y leds, sensor de temperatura.

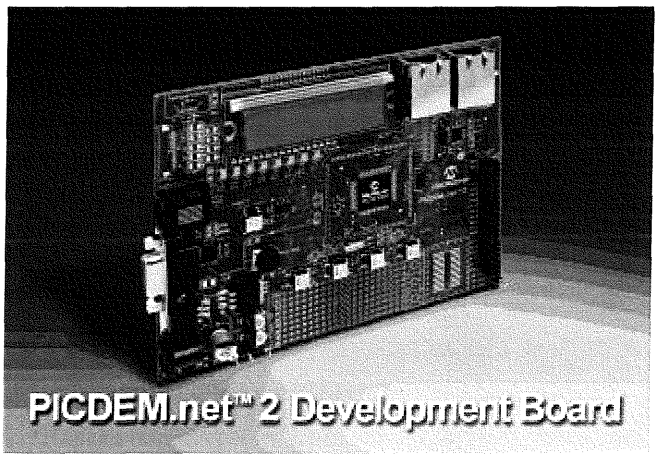

Ilustración 6: PICDEM.net 2 Development Board

Se puede obtener más información sobre el PICDEM.net 2 en la siguiente dirección del fabricante: http://www.microchip.com/stellent/idcplg?IdcService=SS_GET_PAGE&nodeId=1406&dDocName=en028217

El principal inconveniente de esta placa en relación al PIC'SCHOOL, es que los compiladores C libres no ofrecen compatibilidad con el PIC18F97J60. Es también el caso del compilador SDCC, empleado en este proyecto para la programación de la placa. Otras desventajas en comparación con PIC'SCHOOL son:

- No permite la cambiar el microcontrolador incorporado.
- No dispone de generador lógico de frecuencias.
- No dispone de interfaz para el control motores.
- No dispone de protoboard para ampliar sus funcionalidades.

La principal ventaja que ofrece el PICDEM.net 2 es la conexión TCP/IP RJ-45, que nos permite conectar la placa directamente a la red y gracias a esto el servidor central podría comunicarse directamente con la placa. Pero en caso de implantar esta solución seguiría siendo necesario un equipo que procese las imágenes de vídeo.

4.3.2 PIC'Laboratory

El PIC'Laboratory es un sistema de desarrollo dotado de periféricos de entrada y salida que nos permite grabar, depurar y emular programas. Está fabricado en España por la empresa Microsystems engineering.

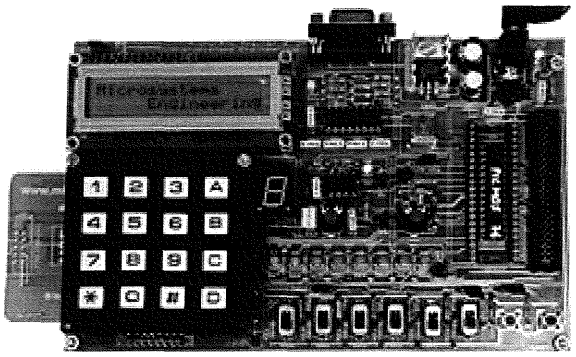

Ilustración 7: PIC Laboratorv

Sus funciones principales son:

1. Servir de entrenador para el rápido aprendizaje en la realización de proyectos con los PIC 16F87X.
2. Servir de herramienta de soporte de los prototipos iniciales de un proyecto para depurar el software control.

Las principales ventajas que nos ofrece es una variedad importante de sensores y periféricos conectados.

Mas información sobre el PIC'Laboratory en la siguiente URL:

http://www.msebilbao.com/tienda/product_info.php?cPath=23_74&products_id=194

Los inconvenientes de este sistema es que se trata de un sistema difícil de ampliar, ya que no podemos añadirle periféricos fácilmente y hay periféricos que no se pueden usar simultáneamente. Solo puede ser utilizado con los microcontroladores de la familia 16F87X, lo que no nos permite escoger el microcontrolador más adecuado a nuestras necesidades.

Son posibles las conexiones USAR pero requieren de un cable especial, los cables serie DB9 estándar no sirven, son incompatibles con el sistema de programación de la placa.

La programación de los microcontroladores solo es posible con un software especifico proporcionado por el fabricante. El microcontrolador se suministra con un sistema cargador de programas, no se puede prescindir de él. Por tanto, no tenemos el control total sobre el microcontrolador.

La principal razón por la que se ha tenido en cuenta este sistema es por la experiencia de haber trabajado anteriormente con él. Sin embargo, tras el análisis de otras opciones se ha descartado esta por verse superada en cuanto a prestaciones por otros sistemas de desarrollo.

4.4 Laboratorio PIC School

El PIC'SCHOOL es un entorno de desarrollo con microcontroladores PIC, que permite desarrollar un proyecto a nivel hardware y software, fabricado con carácter didáctico. Dispone de un amplio y representativo número de periféricos y un módulo Board que nos permite ampliar la cantidad y variedad de los periféricos sin soldaduras. Las posibilidades de ampliaciones son muy superiores a las ofrecidas por las otras placas analizadas.

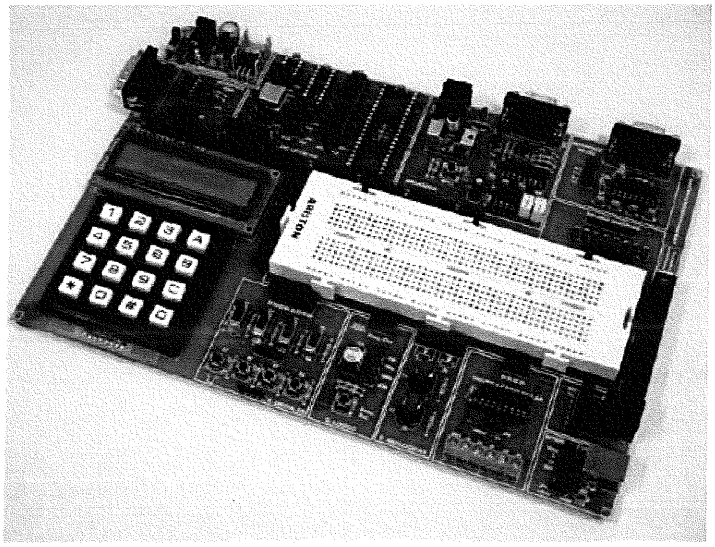

Ilustración 8: Placa de demostración PIC'SCHOOL

Permite trabajar con gran variedad de microcontroladores PIC de las familias 12F, 16F, 18F y dsPIC de 8, 18, 28 y 40 patillas. La programación del microcontrolador se puede realizar sobre la misma placa, gracias al circuito programador y el software que incluye. Gracias a esto se ahorra tiempo en la depuración del programa y se evita daños en las patillas del microcontrolador al no tener que cambiarlo de zócalo para programarlo.

Esta placa cuenta con gran variedad periféricos que no están conectados de forma predeterminada a las patillas del microcontrolador. Las conexiones se realizan con cable rígido de 0.6 mm de grosor, pelando la puntas de los cables y insertando los cables en los zócalos de la placa, como si se tratase de una protoboard. Estas características hacen posible que se pueda probar un proyecto, corregir errores, programar el microcontrolador y modificar las conexiones rápidamente.

Los periféricos incorporados de fábrica son: Pantalla LCD 2x16 caracteres, teclado matricial 4x4, 4 pulsadores, 4 interruptores, 8 diodos LED, interfaz RS-232, interfaz CAN, interfaz USB, conector RJ-11, driver de 6 canales 0.6A, 2 potenciómetros y un generador de frecuencias cuadradas. En la ilustración siguiente se puede observar el esquema del laboratorio PIC'SCHOOL y la localización de los diversos componentes:

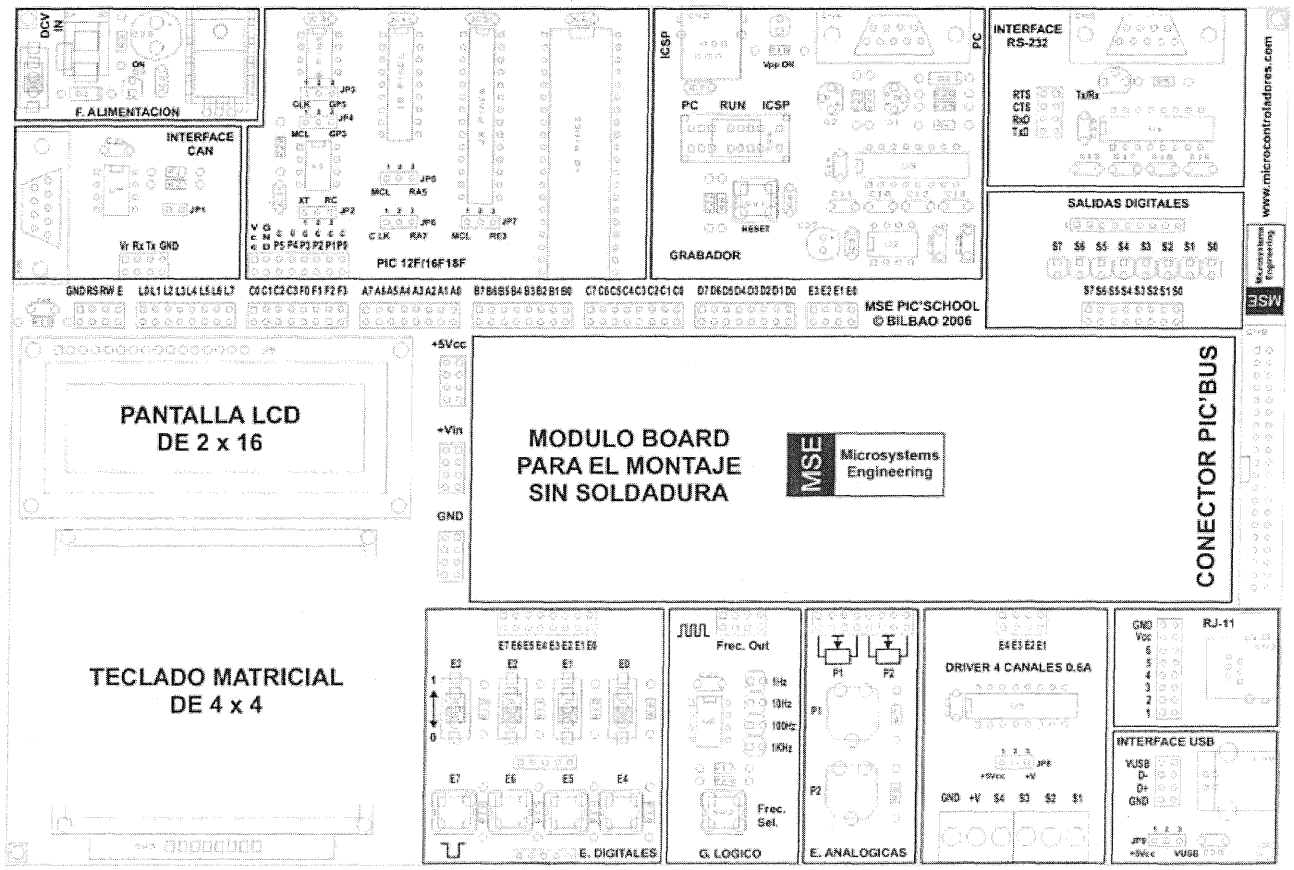

Ilustración 9: Esquema PIC'SCHOOL

En este proyecto se utilizan diversos periféricos que simulan diferentes sensores de una máquina de producción. El microcontrolador captura las señales, las procesa y obtiene datos que comparte con el equipo controlador de sensores. Además se aprovecha la pantalla LCD para mostrar el valor actual de los sensores.

Para conocer más información a cerca del resto de componentes se puede consultar el manual de usuario de la placa:

http://www.msebilbao.com/notas/downloads/Manual%20de%20usuario%20PIC%20School.pdf

4.5 Microcontrolador

Un microcontrolador es un circuito integrado o chip que incluye en su interior las tres unidades funcionales de una computadora: CPU, memoria, unidades de Entrada/Salida, es decir, se trata de un computador completo en un solo circuito integrado.

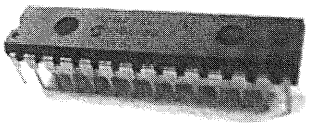

Ilustración 10: Microcontrolador

La función que desempeña dentro del sistema es el control de los periféricos, obtención de datos, comunicaciones. Se encarga de ejecutar el programa de simulación de una máquina de producción. Por tanto, se trata de la parte más importante del simulador.

4.5.1 Selección

4.5.1.1 Compatibilidad con PIC'SCHOOL

El primer paso antes de empezar a programar y probar la placa, consiste en la selección del microcontrolador con el que se trabajará. Hay que tener en cuenta que ha de ser compatible con el laboratorio PIC'SCHOOL y los compiladores de C libres, por tanto, se ha de escoger un microcontrolador PIC de microchip que cumpla las siguientes características:

- La encapsulación ha de ser PDIP (Plastic-Dual-Inline-Package).
- Ha de tener 8, 18, 28 o 40 patillas.
- Las patillas Vss, MCLR, VDD, Vss; han de coincidir con los correspondientes suministros de la placa.
- Se han descartado los dsPIC(16 Bits) por no ser soportados por el compilador de C SDCC, no aparecer en la lista de dispositivos probados con éxito en el PIC'SCHOOL y porque los PIC de 8 Bits son suficientes para realizar la tarea.

4.5.1.2 Prestaciones y funciones

El segundo paso consiste es definir las necesidades en cuanto a prestaciones, una vez definidos todos los sensores. A continuación los requisitos mínimos que ha de ofrecer el microcontrolador seleccionado:

- 1 Puerto de comunicaciones UART para intercambiar información con el sistema de monitorización.
- 1Temporizador de pulsos externos para capturar los flancos de subida del generador de frecuencias.
- 2 Conversores Analógico/Digital para capturar el valor de los potenciómetros.
- 1 Entrada de interrupciones externa para el interruptor de emergencia.
- 11 Entradas/salidas digitales para controlar la pantalla LCD.
- 1 Entrada digital para el interruptor que indica el estado de la máquina (Trabajando o parada).
- 2 Entradas digitales para el control del menú de la pantalla LCD.
- Al menos 14 Kb de memoria de programa para que el microcontrolador pueda alojar el programa.

4.5.1.3 Herramienta de microchip para la selección

Una vez definidas todas estas características, se emplea la herramienta web de filtrado que ofrece microchip para encontrar todos los microcontroladores que ofrecen estas prestaciones. Esta herramienta permite localizar los dispositivos que disponen de unas funcionalidad y características que le indicamos.

El primer paso es escoger la familia de 8 Bits de microchip. Para ello, entramos en la siguiente URL:

http://www.microchip.com/stellent/idcplg?IdcService=SS_GET_PAGE&nodeId=74

El segundo filtro que se ha aplicado es el tipo de memoria, se indicó que se emplearía memoria de tipo flash, ya que son las memorias con más futuro que existen actualmente, están en constante evolución, nos permiten grabarlas y borrarlas miles de veces y son las más rápidas entre las ofrecidas.

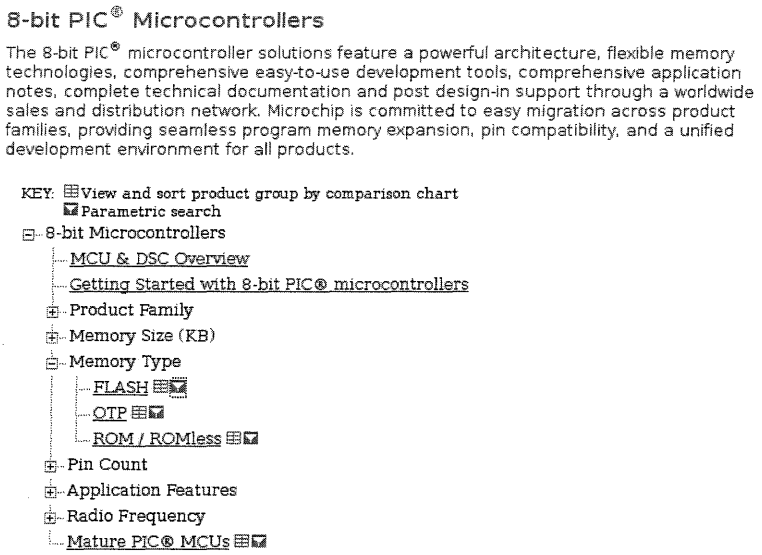

Ilustración 11: Captura de la web de microchip selección de memoria flash

Una vez seleccionado el tipo de memoria flash nos aparece una página con diferentes características para escoger. Teniendo en cuenta las necesidades del simulador ya mencionadas se han seleccionado los siguiente parámetros:

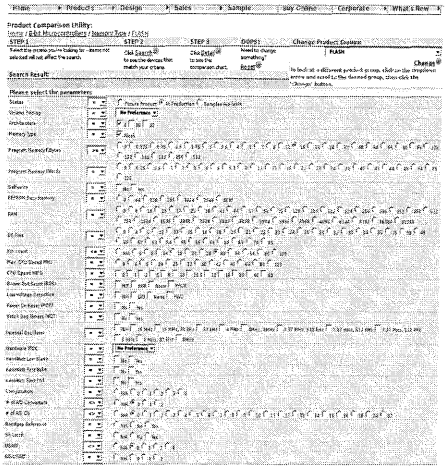

Ilustración 12: Captura de la web de microchip selección de parámetros.

- Program memory Kbytes >= 14. La memoria de programa como ya se ha comentado ha de ser superior a 14 Kbytes.
- Pin count <= 40. El número de PIN del microcontrolador no ha de superar los 40, para ser compatible con el PIC'SCHOOL.
- A/E/UART <> 0. Es obligatorio que tenga al menos 1 puerto UART.
- I/0 Pins >= 21. Se necesitan como mínimos 21 pins de entrada/salida, es la suma que se obtiene de las necesidades ya comentadas.
- Status == In production. Actualmente se están produciendo, por tanto, lo podremos adquirir fácilmente.
- Architecture == 8. Seleccionamos microcontroladores de arquitectura de 8 bits.
- IRDA == No. No es necesario el soporte para conexiones IRDA(Infrarrojos).
- LIN == No. No es necesario el soporte para LIN (Local Interconnect Network).

- Package == PDIP. Indicamos el tipo de encapsulado es PDIP.

Tras el filtrado hecho en la web de microchip quedan 8 posibles candidatos que son: PIC16F767, PIC16F77, PIC16F777, PIC16F876A, PIC16F777A, PIC16F887, PIC16F917 y el PIC18F4450. Por tanto 7 opciones de la gama media y 1 de la gama alta de microchip.

4.5.1.4 Análisis de los candidatos y elección final

A continuación se muestra tabla comparativa donde se aprecian las diferencias entre los diferentes microcontrolados, marcado en más oscuro el modelo escogido:

Modelo	Precio	Patillas	E/S	Canales A/D	Puertos de comunicación	Memoria de Programa
PIC16F767	$3.16	28	25	11	UART MSSP(SPI/I2C)	14 Kbytes
PIC16F77	$3.59	40	33	8	UART MSSP(SPI/I2C)	14 Kbytes
PIC16F777	$3.46	40	36	14	UART MSSP(SPI/I2C)	14 Kbytes
PIC16F876A	$3.40	28	22	5	UART MSSP(SPI/I2C)	14 Kbytes
PIC16F777A	$3.71	40	33	8	UART MSSP(SPI/I2C)	14 Kbytes
PIC16F887	$1.84	40	36	14	UART MSSP(SPI/I2C)	14 Kbytes
PIC16F917	$2.07	40	36	8	UART MSSP(SPI/I2C)	14 Kbytes
PIC18F4450	$2.28	40	34	13	UART	16 Kbytes

La diferencia de precio entre los diferentes modelos es despreciable por ser poco significativa y no ser el precio final del distribuidor. El número de patillas ya es suficiente por los filtros aplicados, la ventaja que ofrece que un chip tenga menos patillas es que ocupará menos espacio. En cuanto al número de E/S cuanto mayor es más información podremos captar y transmitir hacia el exterior. La cantidad de canales A/D indica cuantas patillas se podrán usar como entradas analógicas, tener 14 canales analógicos no quiere decir que sea posible capturar 14 señales analógicas simultáneamente. El número de capturas simultaneas que se pueden realizar lo marca el número de conversores analógico/digital que posea el microcontrolador. La memoria de programa delimita el tamaño del programa que podremos alojar en él.

Tras analizar la tabla, el candidato más adecuado parece ser el microcontrolador de PIC18F4450, porque además de superar en todos estos aspectos al resto de aspirantes, se trata del único PIC de gama alta. Un PIC de gama alta cuenta con varias prestaciones superiores entre

ellas, cuenta con un juego de instrucciones superior (75 frente a las 35 de la gama media) y una frecuencia de trabajo superior (48 Mhz frente a los 20 Mhz de la gama media). Sin embargo, se ha descartado el uso de este microcontrolador por no estar soportado por el compilador SDCC y no ser la única opción válida.

Se han revisado el datasheet[1] del resto de candidatos (Algunos datasheet eran compartidos). Se trata de microcontroladores muy similares y la diferencia más destacable entre ellos el número de canales analógicos. Finalmente se ha decidido que el microcontrolador que se empleará para simular la máquina de producción sea el PIC16F876A. En su elección se ha valorado positivamente el tamaño reducido, cumple todo los requisitos y viene incluido de serie con el laboratorio PIC'SCHOOL.

4.5.1.5 El PIC 16F876A

En este punto se explicarán las principales características del PIC16F876A empleadas en este proyecto, se puede obtener más información consultando el datasheet del microcontrolador: ww1.microchip.com/downloads/en/DeviceDoc/39582b.pdf.

Características principales del microcontrolador:

- Memoria de programa: 14,3 KB.
- Datos de SRAM: 368 Bytes.
- EEPROM (Memoria física no volátil para datos): 256 Bytes.
- Entradas/Salidas digitales: 22.
- Entradas Analógicas: 5.
- Soporte para UART.
- 3 Temporizadores.
- 1 Puerto de interrupciones externas.

1 Un **datasheet** es un documento que resume el funcionamiento y otras características de un componente (por ejemplo, un componente electrónico) o subsistema (por ejemplo, una fuente de alimentación) con el suficiente detalle para ser utilizado por un ingeniero de diseño y diseñar el componente en un sistema.

Hay que tener en cuenta el siguiente esquema ya que no es posible utilizar algunas de las prestaciones simultáneamente, por ejemplo, no es posible utilizar el puerto digital RC7 y el canal transmisión UART.

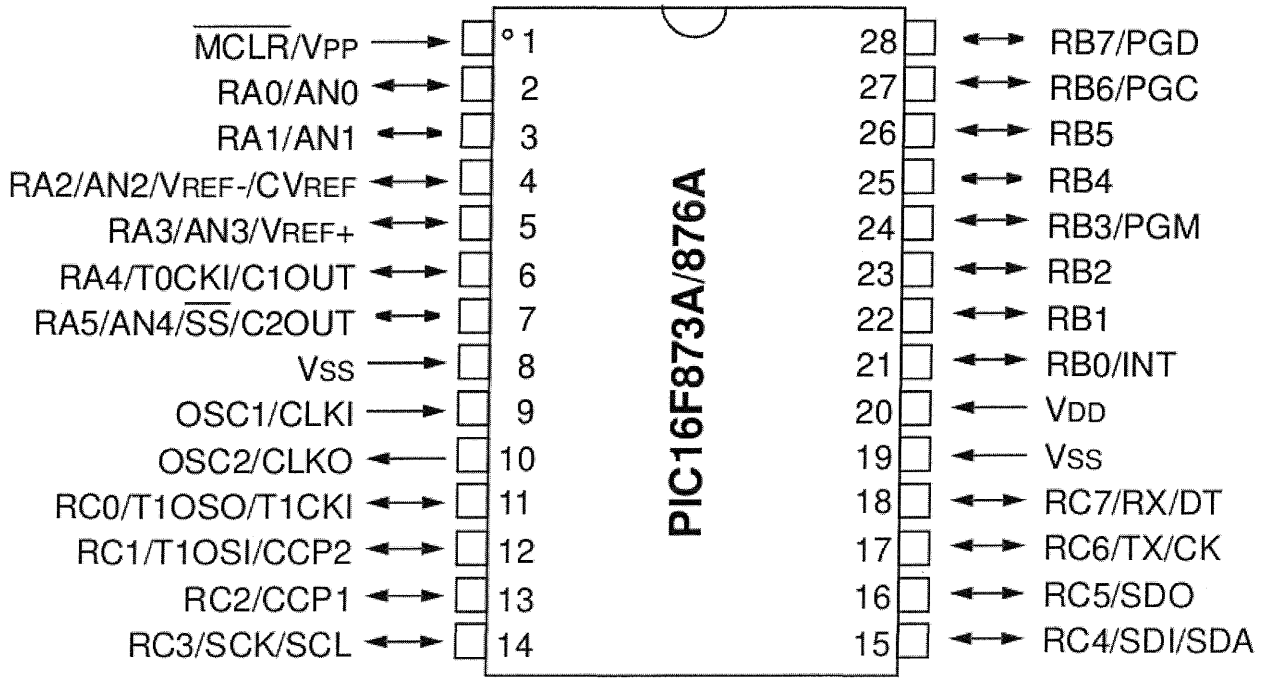

Ilustración 13: Esquema del PIC 16F876A

4.6 El compilador SDCC, lenguaje C

Para trabajar con microcontroladores lo más normal es programarlos en lenguaje ensamblador o en lenguaje C. El lenguaje ensamblador permite tener un control más óptimo del hardware que el lenguaje C y los programas son más eficientes. Pero el mayor inconveniente es que programar en ensamblador es lento, los programas resultan poco legibles y más complicados que con C, en caso de errores son más difíciles de localizar. Teniendo en cuenta las ventajas que ofrece C frente al lenguaje ensamblador y que el programa será relativamente grande, se ha decidido que se programará en C.

El contenido de la memoria de programa ha de ser un programa en código máquina para que el microcontrolador lo pueda interpretar, para poder producir código máquina a partir de un programa escrito en C es necesario el uso de una herramienta llamada compilador.

4.6.1 Selección del compilador

Existe gran variedad de compiladores C para PIC, con amplias librerías de funciones pero la mayoría requieren la compra de una licencia para su uso con animo de lucro. Lista de los más populares compiladores para C: CCS, Hi-Tech, MicroC (http://www.mikroe.com/en/compilers/mikroc/pic/), SDCC (http://sdcc.sourceforge.net/) y otros compiladores que ofrece microchip (http://www.microchip.com/stellent/idcplg?IdcService=SS_GET_PAGE&nodeId=81). Hay que tener en cuenta que dos compiladores no tienen porque ser capaces de interpretar un mismo código o de traducirlo a código máquina de la misma forma.

4.6.1.1 CCS

Se trata de un compilador C que viene acompañado de un gran número de librerías que nos permiten trabajar de una manera cómoda con el microcontrolador, además de ofrecernos soporte para una amplia gama de periféricos.

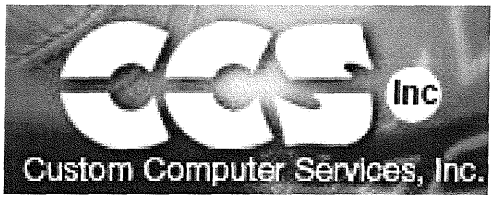

Ilustración 14: Logotípo CCS

También es destacable la amplia comunidad de usuarios que trabajan con este compilador, por lo que resulta fácil solucionar las dudas y encontrar información sobre él. Se trata de un lenguaje muy preprogramado, ya que facilita mucho la programación y no es necesario preocuparse mucho por el funcionamiento del hardware como sí pasa con otros compiladores C. Permite trabajar con PIC de todas las familias PIC10, PIC12, PIC16, dsPIC, PIC24 y PIC32.

Para poder trabajar con este compilador es necesario trabajar en entorno Windows, y el precio de la versión más económica se sitúa en los 350$. Más información a cerca de de este compilador en: http://www.ccsinfo.com/content.php?page=compilers

4.6.1.2 Hi-Tech para microchip

Se trata de otro compilador de C con soporte para microchip, también con muchos usuarios pero no dispone de tantas librerías como CCS para trabajar con él. Es un lenguaje menos prefabricado que el CCS, se ha de conocer mejor el hardware, por tanto es un lenguaje más complejo (Por ejemplo: Cuando programamos con Hi-Tech es necesario conocer todos los registros del PIC para trabajar con un temporizador, con CCS no es necesario, solo tenemos indicar que queremos hacer con el temporizador mediante una función). Permite trabajar con todas las familias de microcontroladores PIC10/14/16/18, dsPIC3x/PIC24 y PIC32.

Ilustración 15: Logotipo Hi-Tech

Dispone de un compilador gratuito el HI-TECH PICC-Lite Compiler, que permite trabajar con algunos de los más populares PIC, pero la mayoría de ellos con limitaciones de tamaño de programa por lo que esta versión del Hi-Tech no resulta útil para este proyecto (Por ejemplo: 16F877 2 RAM banks, 2K program memory supported). La versión completa más económica que permite trabajar con el microcontrolador elegido cuesta 995.00 $.

http://microchip.htsoft.com/

4.6.1.3 Compiladores de Microchip C18, C30, C32 .

Ilustración 16: Logotipo de Microchip

La propia empresa microchip nos ofrece compiladores C para las familias PIC18, PIC24, dsPIC, y PIC32. No ofrece compiladores C para la gama media y baja, si se eligiese este compilador habría que trabajar con el PIC18F4450.

En cuanto a la programación es parecido al Hi-tech, pero la principal ventaja que ofrece es el entorno de trabajo MPLAB, muy práctico para desarrollar proyectos. Cuenta con una documentación muy detallada accesible desde la propia web de microchip, cuenta con una amplia comunidad de usuarios y está desarrollado por el propio fabricante de los chip. Además, cuenta con una versión para uso de estudiantes pero no es tan óptima como la versión comercial. Los principales inconvenientes por los cuales no se ha escogido son: solo es compatible con el sistema operativo Windows y solo la versión de estudiantes es gratuita.

4.6.2 El compilador SDCC

Se trata de un compilador C código abierto (Open-source) con licencia GNU, en otras palabras se trata de un compilador gratuito. Ofrece soporte para la mayoría de PIC de gama baja y media, también para algunos PIC de gama alta (PIC18). Además ofrece soporte para otras marcas y modelos de microcontroladores: Intel 8051, Maxim 80DS390, Zilog Z80 y el Motorola 68HC08. No es fácil encontrar documentación sobre el uso de este compilador para los PIC, y tampoco ejemplos. Sin embargo para utilizarlo es suficiente con conocer el lenguaje C y el hardware con el que se trabaja. No cuenta con una amplia comunidad de usuarios como los otros compiladores.

Una interesante ventaja es que se puede trabajar con él en múltiples plataformas Linux, Windows y Mac OS. Se trata de la opción elegida porque cumple todos los requisitos y es el que más se adapta a las necesidades y objetivos de este proyecto.

4.7 El programa del simulador

El objetivo principal del programa es recoger información de unos sensores conectados al microcontrolador y transmitirlos vía UART cuando se le solicitan. El programa capta el valor de los sensores de diferentes, constantemente.

4.7.1 Descripción de los periféricos empleados

La placa PIC'SCHOOL cuenta con diversos periféricos que se pueden ampliar con la ayuda de la protoboard que incorpora, a continuación se describen los dispositivos empleados y el uso.

4.7.1.1 Potenciómetros

El potenciómetro o resistencia variable es un dispositivo electromecánico, al cual se le puede variar el valor de su resistencia. Esta formado por una resistencia de valor fijo y un contacto deslizante (Cursor) que la divide eléctricamente.

Mediante la rotación del cilindro rojo del potenciómetro de la placa, desplazamos el cursor y modificamos la intensidad y tensión que circula por el circuito.

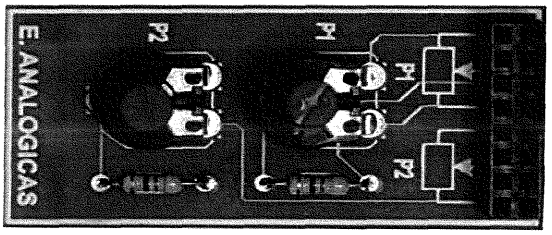

Ilustración 17: Potenciómetros PIC'SCHOOL

Los potenciómetros están conectados a una diferencia de potencial de 5V (0V-5V) entre sus extremos, que es equivalente a tensión de referencia por defecto de los convertidores analógicos digital del microcontrolador. La principal ventaja de esta configuración es que no es necesario que proporcionemos tensión de referencia positiva y negativa, lo que nos permite liberar dos patillas, que en caso contrario estarían dedicadas a esta tarea.

Se utilizan los dos potenciómetros incluidos en el PIC'SCHOOL (P1 y P2). El potenciómetro P1 simulará un termómetro, la escala de temperatura es de 0 a 255 ºC. P2 simulará el medidor de capacidad de un depósito, el valor devuelto es el porcentaje de llenado del deposito.

Como se puede observar en la Ilustración 15, el cursor del potenciómetro P1 está directamente conectado a la entrada RA0 y P2 a RA1.

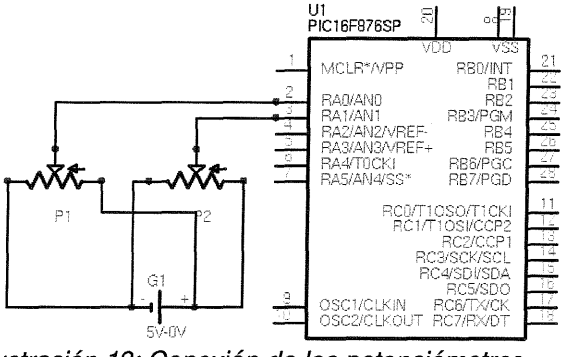

Ilustración 18: Conexión de los potenciómetros.

4.7.1.2 Generador lógico

El generador lógico incluido en la placa se trata de un circuito electrónico basado en el dispositivo SYM10AA que es capaz de generar ondas cuadradas simétricas y cuadradas a frecuencias 1 Hz, 10 Hz, 100 Hz y 1KHz.

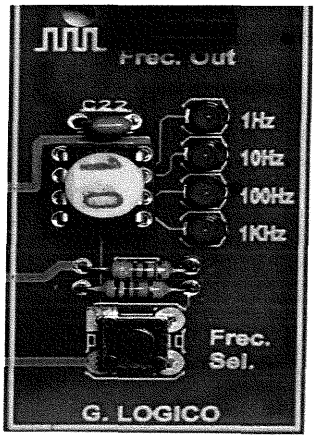

Ilustración 19: Generador lógico

Pulsando el pulsador de selección de frecuencia se puede escoger la frecuencia de las ondas generadas. El circuito dispone de LEDs que nos indican la frecuencia de salida.

El generador lógico simula un tacómetro[2] y su salida son impulsos con forma de onda cuadrada, simula la velocidad de giro del eje de un motor de la máquina monitorizada.

En la ilustración 17 se puede comprobar que la salida del generador lógico se encuentra conectada a la entrada T1CKI del microcontrolador.

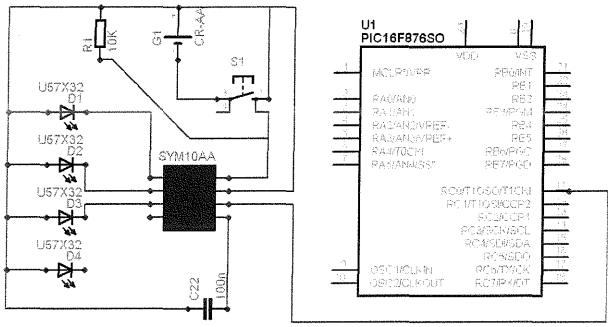

Ilustración 20: Esquema Generador Lógico

4.7.1.3 Interruptor de emergencia e interruptor de estado

El entrenador PIC'SCHOOL dispone de 4 interruptores y 4 pulsadores, con la finalidad de que se empleen como entradas digitales.

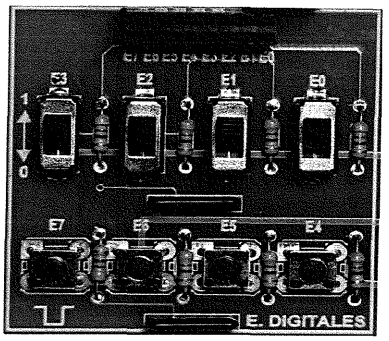

Ilustración 21: Interruptores y pulsadores PIC'SCHOLL

La diferencia entre estos elementos es que los interruptores disponen de dos posiciones estables, mientras que los pulsadores disponen de una posición estable y otra temporal que solo se mantiene mientras el usuario lo mantiene presionado.

2 Instrumento de medición de la velocidad del giro de un eje.

Se emplea el interruptor E1 para simular el botón de encendido (1) / apagado (0) de la máquina, el E0 de interruptor de emergencia y los pulsadores E5 y E4 como teclas anterior y siguiente del menú que aparece por la pantalla LCD.

Los elementos de conmutación de circuito se encuentran conectados a las siguiente entradas digitales del microcontrolador, E1 a RC5, E4 a RA4 y E5 a RA5. E0 se encuentra conectada a RB0/INT para que produzca una interrupción cada vez que se cambie el estado.

4.7.1.4 La pantalla LCD

Se emplea la pantalla LCD incorporada en el PIC'SCHOLL, para mostrar el valor de los sensores, la fecha, hora actual y otros mensajes al usuario. Se trata de una pantalla de cristal líquido, monocromática, compatible con HD44780 y que dispone de los filas que permiten mostrar de 16 caracteres alfanuméricos cada una.

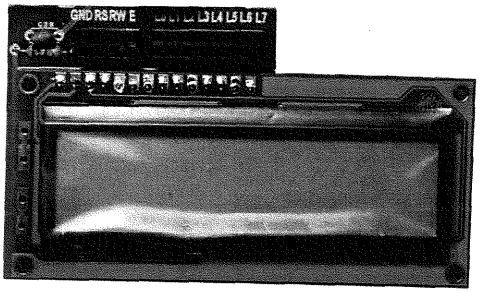

Ilustración 22: Pantalla LCD

Dispone de las siguiente conexiones: Un bus de 8 bits para transferencia de datos, 3 bits de control y 3 entradas de alimentación.

Como se puede observar en la Ilustración 20, el bus de datos de la pantalla se encuentra conectado al puerto B del microcontrolador a excepción de la patilla D0 del LCD que se encuentra conectada a la puerta digital RC4.

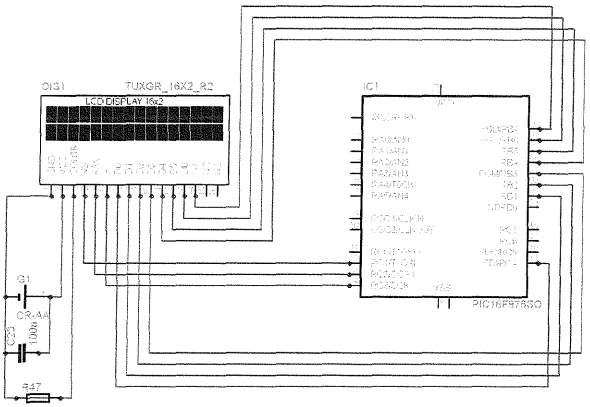

Ilustración 23: Esquema conexión pantalla LCD

4.7.1.5 El puerto serie RS-232 (UART)

La interfaz RS-232 se basa en el chip MAX-232 y el conector hembra DB9, se emplea para el intercambio de datos binarios en serie.

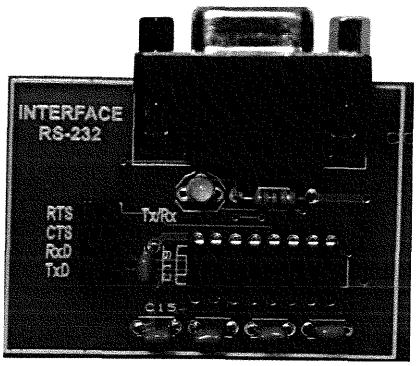

Ilustración 24: Interfaz RS-232

Se emplea este dispositivo para transmitir los valores obtenidos de los sensores al PC receptor de sensores. En la demostración de este proyecto, el PC receptor es un portátil que no dispone de puerto serie, para salvar este inconveniente se ha adquirido un adaptador Serie-USB, por tanto, el PIC'SCHOLL se encuentra conectado a la entrada USB del portátil. El inconveniente del adaptador es que no proporciona suficiente tensión para programar el Microcontrolador.

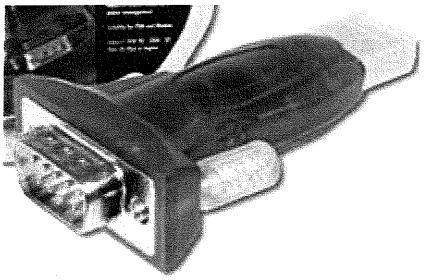

Ilustración 25: Adaptador USB-Serie

Para la comunicación con el PC se emplean 2 puertas digitales del microcontrolador Tx (Transmisión) y Rx (Recepción).

4.7.2 Aspectos generales de la programación en C de microcontroladores

El objetivo de esta sección es explicar las peculiaridades de la programación de microcontroladores y la forma en que se realizan las operaciones empleadas en este proyecto. Para entender sin problemas el siguiente punto es necesario estar familiarizado con los lenguajes de programación.

4.7.2.1 Registros del microcontrolador

Durante la programación del microcontrolador es necesario trabajar con los registros, los registros son espacios de memoria de 8 Bits (1 Byte). Hay diferentes tipos de registros.

- Los registros de configuración: Como por ejemplo, el registro TRIAS que define que patillas del puerto A son de entrada y cuales de salida. Todos estos registros y cada uno de sus bits están definidos en el Datasheet de este microcontrolador.
- Otra clase de registros son los utilizados para almacenar datos, como por ejemplo el registro, PORTA donde se guarda el valor recibido por las entradas digitales del puerto A o el valor que se transmite.
- Otra tipo de registros son los de propósito general que se emplean para guardar el valor de las variables de los programas que ejecutan.

La memoria de registros es de tipo RAM, por tanto, se borra cada vez que el microcontrolador se reinicia. Si queremos guardar el valor de una variable tras un reinicio tendremos que emplear los registros de la memoria EEPROM.

4.7.2.1.1 Como asignar un valor a un registro

Para modificar un registro en C, basta con usar el nombre del registro que se va a modificar y asignarle en nuevo valor. Por ejemplo, si se ejecuta la siguiente línea: "PORTB=0x21". El valor binario del registro PORTB pasa a ser 00100001.

Sin embargo, si únicamente se pretende modificar el valor de algunos bits del registro manteniendo el resto, es necesario emplear máscaras.

4.7.2.1.2 Máscaras

Para modificar el valor algunos bits de un registro, sin modificar el resto emplearemos las máscaras y la operación AND o OR bit a bit (Símbolo "&" y "|" respectivamente). La máscara es un grupo de bits del tamaño del registro al que se aplica. En este caso se aplica bit a bit, el bit 0 de la máscara opera contra el bit 0 del registro, el bit 1 de la máscara contra el bit 1 del registro y así sucesivamente. Los resultados de la operación AND y OR se pueden observar en la Ilustración 26.

Entrada		AND	OR
A	B	A&B	A|B
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	1

Ilustración 26: AND

En el caso del AND, el bit n de la máscara igual a 0 obliga a que el bit n del registro tome como valor 0. Un bit n igual a 1 en la máscara mantiene el valor el correspondiente al bit n en el registro. Un ejemplo práctico de una máscara que modifica a 0 bits de un registro es el siguiente: "TRISA=TRISA&0xFC". Este comando establece el valor de los dos bits más a la derecha de TRISA a 0. En la Ilustración 27 aparece un ejemplo de la operación en la que TRIASA es igual a 1101110.

Registro	Salida
TRISA	1101110
Mascara	11111100 (0xFC)
Resultado	1101100

Ilustración 27: And bit a bit

En cuanto a la operación OR el bit de la máscara a 0 mantiene el valor del bit en el registro y el 1 modifica el valor del bit del registro a 1. En ejemplo se emplea la misma mascara y valor para TRIASA pero se modifica el operador "TRISA=TRISA|0xFC".

Registro	Salida
TRISA	1101110
Mascara	11111100 (0xFC)
Resultado	1111110

Ilustración 28: Or bit a bit

4.7.2.2 Interrupciones

Una interrupción es una señal recibida por el procesador del microcontrolador que le indica que debe interrumpir el curso de ejecución actual del programa y pasar a ejecutar código específico para tratar esta situación.

Cuando se produce una interrupción el microcontrolador pasa a ejecutar el código del registro 0004h de memoria de programa, en el programa implementado en este proyecto la instrucción introducida en dicha posición de memoria, llama a la rutina de atención a las interrupciones general. Esta rutina determina a partir de los registros Flag (Bandera, como por ejemplo INTF), que interrupción se ha interrumpido el flujo[3].

Si se quieren activar las interrupciones o bien deshabilitarlas en un momento dado, se ha de asignar el valor 0 o 1, respectivamente, al registro GIE (General interrupt enable). También esta permitido habilitar o deshabilitar interrupciones concretas, mediante los registros enable de

3 Se denomina flujo de ejecución al conjunto de acciones que realiza un programa

cada interrupción que aparecen en la siguiente ilustración.

FIGURE 14-10: INTERRUPT LOGIC

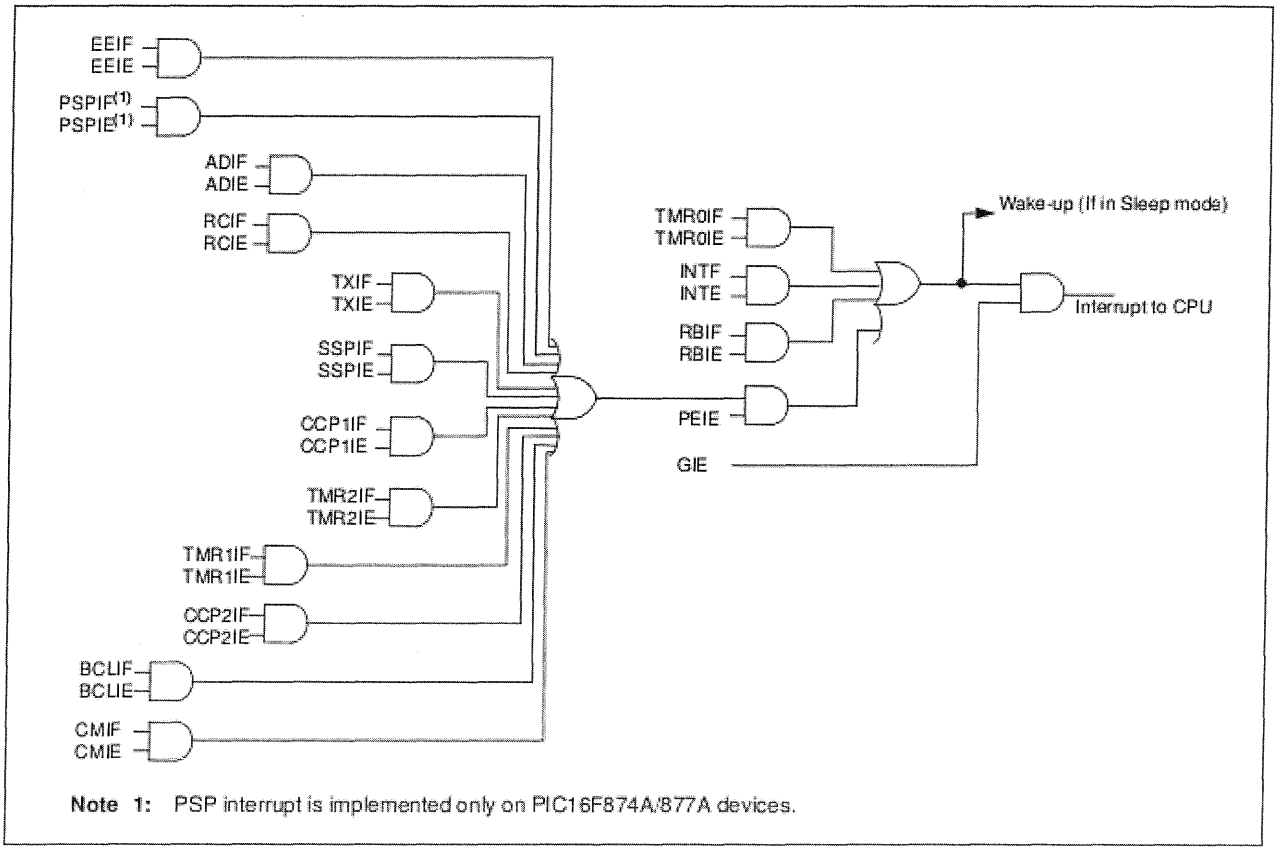

Ilustración 29: Esquema interruciones PIC16F876A

Se puede llegar a la situación en que durante la atención una interrupción, se produce otra, en este caso se interrumpe el flujo de atención a la primera interrupción y se pasa atender la segunda.

El principio seguido para la programación de las rutinas de atención a las interrupciones consiste en hacer funciones ligeras para devolver lo antes posible el flujo al programa principal y dejar el camino libre para la próxima interrupción.

4.7.2.3 Efecto rebote en interruptores y pulsadores

Los pulsadores e interruptores son elementos mecánicos básicos, por lo tanto no ofrecen un cambio de estado de 0 a 1 o viceversa perfecto. Durante una pulsación un interruptor o un pulsador aparecen falsas pulsaciones, nombradas rebotes o Button Bounce, representadas en la zona interior del recuadro gris de la Ilustración 31. Este fenómeno se produce cuando se reduce mucho la distancia entre los contactos, en ese momento, la corriente puede conducir de forma irregular a través del aire que separa los contactos, hasta que el contacto es total.

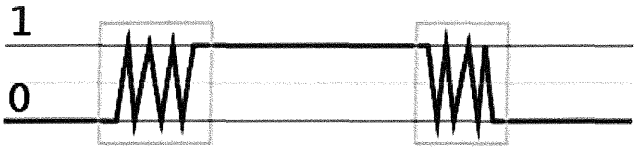

Ilustración 31: Representación del efecto rebote

A continuación se explica el ejemplo de secuencia de pulsación de la Ilustración 30:

1. El interruptor se encuentra en reposo, no ha sido pulsado, se encuentra abierto, el valor de salida es 0.
2. El interruptor está siendo pulsado, la proximidad entre los contactos escasa, lo que provoca la formación de un arco eléctrico que cierra el circuito, el valor de salida es 1.
3. Durante la pulsación del botón el arco eléctrico se rompe, dejando el circuito abierto, el valor de salida es 0.
4. El contacto de los dos extremos es total y el circuito queda del todo cerrado, el valor de salida es 1. El proceso se repite en sentido contrario cuando se suelta el pulsador.

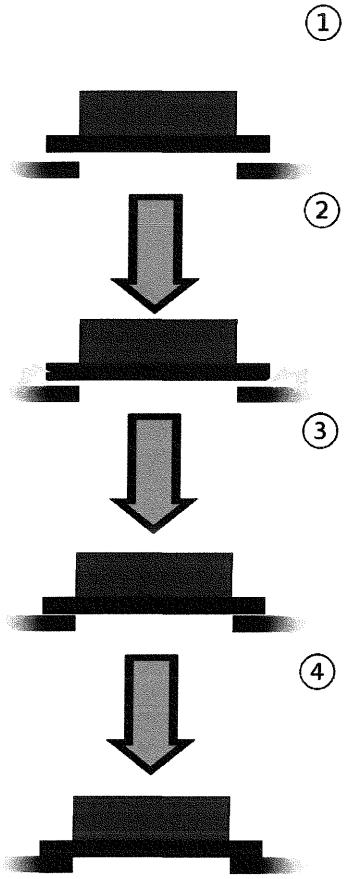

Ilustración 30: Secuencia de pulsación de un pulsador

La velocidad del microcontrolador PIC es suficiente alta para verse afectado por este efecto. El rebote puede provocar errores si no se controla, como en el ejemplo descrito a continuación, en el que se detectan varias pulsaciones cuando solo se ha producido una. Esto ocurre en algunos casos, cuando varias lecturas se producen en la misma zona de Bounce, como en la Ilustración 32, donde los recuadros azul representan las lecturas. En este caso, la primera lectura devuelve 0 al no estar pulsado el pulsador, la segunda lectura devuelve un 1, la tercera captura devuelve un 0, por tanto, el programa ya interpreta una pulsación. La última lectura

marcada devuelve un 1, por tanto si el programa no tiene en cuenta el efecto de rebote detectará dos pulsaciones en esta captura.

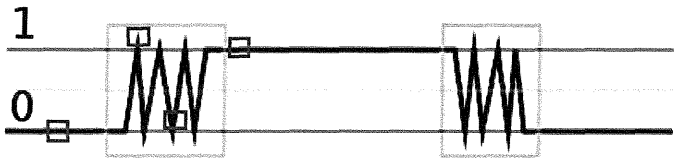

Ilustración 32: Lecturas del pulsador

Para evitar pulsaciones fantasma existen soluciones hardware y soluciones software. Las primeras suelen consistir en la colocación de un condensador o la utilización de pulsadores especiales.

En los siguientes apartados se describen dos soluciones software al problema:

4.7.2.3.1 Capturas periódicas

Esta solución consiste en la consulta del valor de interruptor cada t tiempo, se definen los parámetros a tener en cuenta en una pulsación a continuación.

- t= Tiempo que trascurre entre una lectura del interruptor y la siguiente. Se trata de una constante durante todo el programa. Tiempo transcurrido entre las líneas naranja.
- tr= Se trata del tiempo máximo que puede llegar a durar la zona de rebote.
- tm= Tiempo que dura la pulsación más rápida, desde que se termina la zona de rebote de pulsación hasta que empieza la zona de rebote por soltar el pulsador.
- ts = Tiempo mínimo que transcurre entre una pulsación y la siguiente.

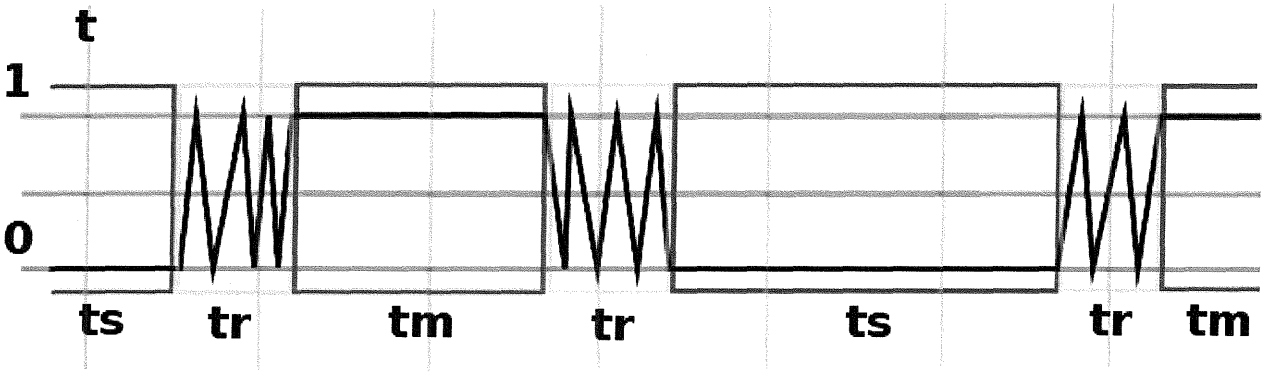

Ilustración 33: Señal transmitida por un pulsador

El tiempo entre captura y captura ha de cumplir que t>tr, t<tm y t<ts.

La complejidad que plantea este método reside en que hay que controlar el tiempo que pasa entre lectura y lectura, para impedir que sobre pase el intervalo marcado. Es fiable pero es necesario hacer pruebas para adaptar los tiempos. Se puede implementar empleando un contador que genere una interrupción cada t tiempo.

4.7.2.3.2 Tiempo de pulsación

Se trata de otro método muy similar consiste en realizar consultas periódicas del estado del pulsador (polling[4]) en tiempo inferior o igual a tt/2, donde tt es el tiempo que dura la pulsación más corta que puede hacer el usuario, es decir "tt = tr inicial + tm + tr final". A partir de ese momento se espera un periodo de tiempo superior al tiempo de rebote que no sobrepasa tt/2. Transcurrido este tiempo se vuelve a capturar el valor de la entrada y si el estado del pulsador se mantiene desde el cambio de estado anterior se acepta la pulsación, en caso contrario se desestima. Este es el método aplicado a todos los pulsadores e interruptores empleados en el proyecto porque es más fácil de implementar y consume menos recursos.

4.7.3 El programa principal y sus diagramas de flujo

Antes de empezar a programar y crear el programa para el microcontrolador hay que conocer a fondo como programar los diferentes módulos microcontrolador, para que interaccionen con los periféricos y el mundo exterior. Para cumplir este punto, se hicieron prácticas con las diferentes funcionalidades que se han implementado. Toda la información acerca del microcontrolador se ha conseguido consultando el Datasheet del PIC16F846A.

Uno de los principios que se han seguido para la elaboración del programa es la no interrupción del flujo de programa con consultas bloqueantes. Se ha implementado así, para evitar la perdida de tiempo que suponen las esperas pasivas, que detienen el flujo del programa hasta que se obtiene el resultado de la consulta. Para hacer esto posible, el resultado de las consultas se obtiene a través de interrupciones o mediante consultas periódicas no bloqueantes y el tiempo ahorrado se emplea para procesar otros resultados ya disponibles y tareas.

4 Operación de consulta constante, generalmente hacia un dispositivo hardware, para crear una actividad síncrona sin el uso de interrupciones.

A continuación, el diagrama de flujo del programa principal donde se esquematiza el funcionamiento del programa y también aparecen las interrupciones que pueden variar temporalmente el flujo normal del programa (Interrupción) .

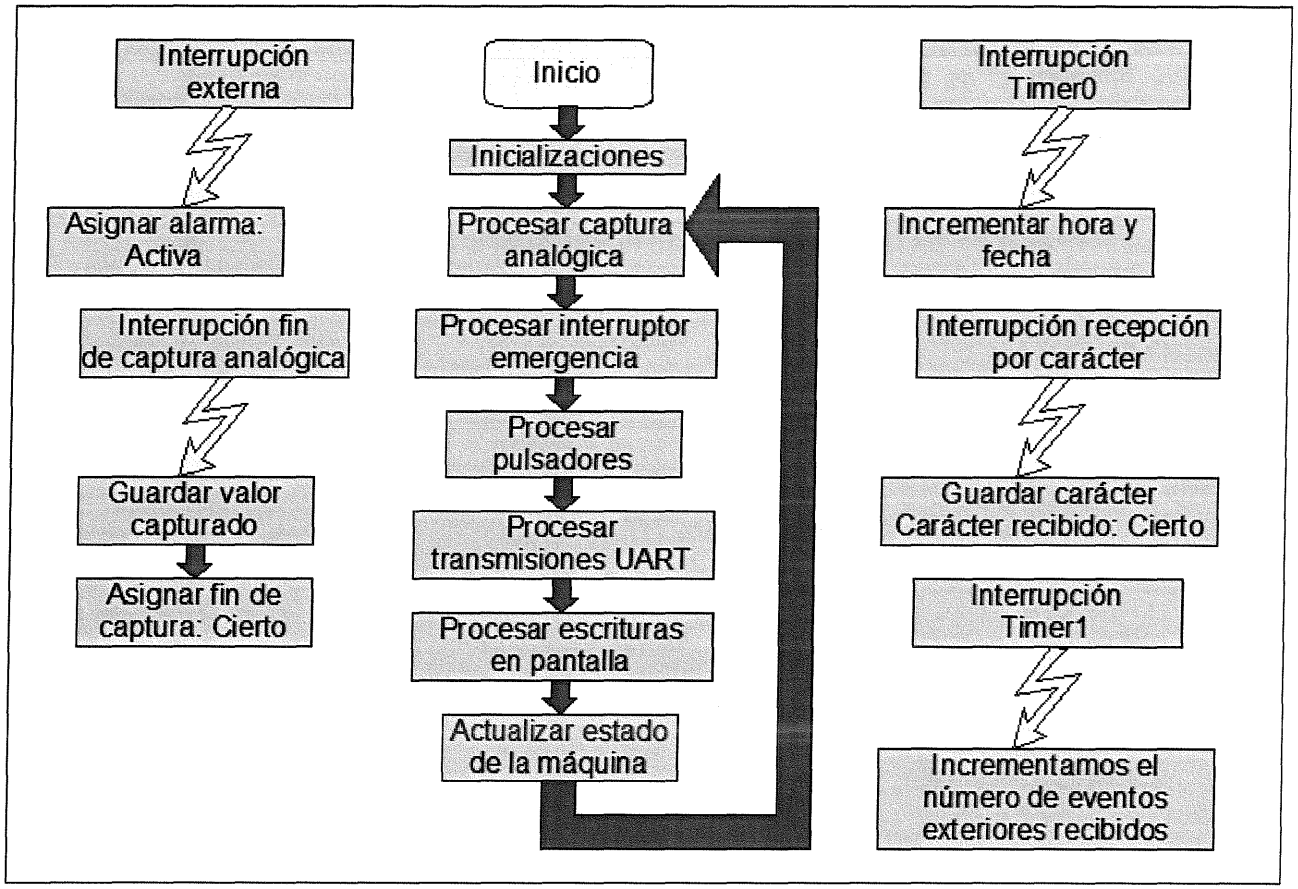

Ilustración 34: Diagrama de flujo del programa principal

El programa principal esta formado por un bucle infinito en el que se ejecutan seis procesos que actualizan el valor de los sensores periódicamente y transfieren la información al equipo capturador de sensores.

A continuación se describe un ejemplo practico de flujo de ejecución del programa: Nos situamos al inicio del bucle principal, el primer paso consiste en dar paso a la rutina que controla la captura analógica. En la rutina se detecta que la última captura analógica no ha acabado, por tanto, se devuelve el flujo al programa principal sin esperar a que acabe la captura. Una vez pasamos a procesar las transmisiones USART se da el caso de que la última transmisión a terminado, en consecuencia la rutina USART carga el siguiente carácter a transmitir y sin esperar a que se transmita devuelve el flujo al programa principal.

4.7.3.1 Inicializaciones

En esta sección se preparan todas las variables de entorno y componentes para empezar a trabajar con el bucle principal del programa, el orden de las inicializaciones no es relevante pero sí es importante no modificar nada de lo anteriormente configurado.

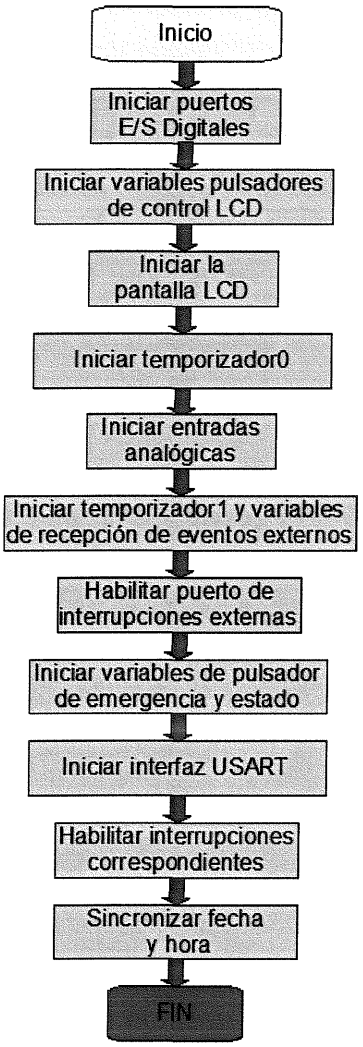

Ilustración 35: Diagrama inicializaciones

Por ejemplo, si se han configurado las patillas 1 y 2 de un puerto como salida y en otra sección de inicializaciones queremos modificar las patillas 5 y 7 de ese puerto hemos de tener cuidado de no modificar el estado de las patillas modificadas anteriormente 1 y 2. Para conseguir esto recurriremos a las máscaras de bits ya comentadas.

Iniciar puertos E/S Digitales: Se asigna el valor correspondiente a cada puerta digital según corresponda, 1 (Entrada) o 0 (Salida) en cada bit del registro TRIS de la puerta. Por ejemplo:

· TRISA4=1; Asignamos al bit 4 del registro TRISA el valor 1, con lo que asignamos como entrada la puerta 4 del puerto A (RA4).

· TRISC=0xF1&TRISC; Asignamos a los bits 3, 2 y 1 el valor 0, el resto de registros mantienen su valor. Con esta operación configuramos como salida las puertas RC1, RC2 y RC3.

Iniciar variables pulsadores de control LCD: Se inician las variables de los pulsadores que controlan el menú LCD, de forma que el valor de iniciación es "*Ningún botón pulsado*".

Iniciar la pantalla LCD: Se inician las variables de la pantalla LCD para que nada más empezar muestre el menú 0.

Iniciar temporizador 0: Se programa y activa el temporizador 0 para que produzca un interrupción cada 10 Ms, asignando una escala 1:256 al preescaler. Se emplea la frecuencia funcional del microcontrolador como entrada del temporizador. Se inicia el contador del timer 0 (TMR0) a 217 para esperar 39 tics de reloj antes de provocar la interrupción. Por defecto, se inician todas las variables relativas a la fecha y hora a 0.

Iniciar entradas analógicas: En este punto se preparan los registros para empezar con las capturas analógicas de los puertos RA0 y RA1. Se activa el conversor analógico y las interrupciones que indican el fin de conversión.

Iniciar el temporizador 1: En este punto se inician los registros que controlan el temporizador 1, para que trabaje con los flancos de subida que recibe del exterior, se configurará de forma que cada vez que reciba un evento exterior provoque una interrupción. También es necesario en este paso iniciar las variables que controlan el número de eventos trascurridos.

Habilitar interrupciones externas: Para habilitar las interrupciones externas es necesario configurar la patilla RB0 como entrada de interrupciones, para que cada vez que reciba un flanco de subida provoque una interrupción que nos informe de la pulsación del interruptor de emergencia.

Inicializar variables de pulsador de emergencia y estado: Las variables de pulsador de emergencia y de estado son iniciadas debidamente asignando el valor de los pulsadores en el momento del inicio del simulador.

Iniciar interfaz USART: Se ha configurado la interfaz USAR con los parámetros siguientes, velocidad 9600 baud, 8 bits, sin bit de paridad, sin control de flujo y transmisión asíncrona. Cada vez que se recibe un carácter se recibe un interrupción.

Habilitar interrupciones corresopondientes: Se activan las interrupciones que se emplearán (Timer0, timer1, INT, etc) y se permite la recepción de interrupciones por parte de la CPU, activando del bit GIE (Global Interrupt Enable bit).

Sincronizar fecha y hora: Para tener la hora actualizada automáticamente sin necesidad introducirla, se sincroniza la placa con el PC capturador empleando el puerto USART.

4.7.3.2 Capturas analógicas

El microcontrolador PIC16F876A, como la mayoría de controladores de la familia, solo dispone de un conversor analógico/digital (CAD), por lo que es necesario hacer las capturas de 1 en 1, pese a disponer de 5 entradas analógicas.

El proceso de captura analógica es secuencial, captura el valor de las entradas analógicas empezando por AN0 hasta ANx. (x=1). En el caso de este simulador recibimos dos entradas analógicas AN0 (Temperatura) y AN1 (Nivel del deposito).

El programa prepara al microcontrolador para producir una interrupción cada vez que el CAD termina una conversión. La rutina de atención a la interrupción correspondiente se encarga de salvar en la variable correspondiente el resultado de la última captura.

4.7.3.3 Procesar el interruptor de emergencia

El interruptor de emergencia E0, se encuentra directamente conectado a la entrada de interrupciones externas INT, con esto se consigue que en el momento que se activa el interruptor se produzca una interrupción y se pase a ejecutar el código de atención a las interrupciones.

Para tener actualizado en todo momento el estado de este interruptor al iniciar el programa se lee el estado del interruptor y se actualizar el valor de la variable de estado del interruptor (Variable alarma). Cuando se recibe una interrupción externa el valor de alarma pasa a "Activo" y se desactivan las interrupciones externas para evitar los rebotes producidos por los interruptores al cambiar de estado. Para detectar que el botón de emergencia ya no está pulsado, leemos periódicamente el valor del bit 0 de la puerta B. Cuando deja de estar pulsado reactivamos las interrupciones externas y cambiamos el valor de alarma a 0.

4.7.3.4 Procesar pulsadores

En este punto se tratan los pulsadores de menú de la pantalla que permiten conocer en tiempo real el valor de los sensores. La única complejidad de la programación de estos pulsadores, como ocurre con el resto de interruptores, es evitar recibir falsas pulsaciones por el efecto rebote. Para cambiar el menú se modifica el valor de la variable global menulcd cuyo valor es el número de menú que se imprime por la pantalla.

4.7.3.5 Procesar transmisiones USART

La placa funciona como un servidor y el PC capturador como un cliente. El cliente realiza una petición de datos (¿Temperatura?), el servidor responde con los datos solicitados (Temperatura 80ºC).

Para realizar correctamente el intercambio de datos entre un equipo y el otro es necesario sincronizarlos correctamente. Para la recepción de datos se emplean interrupciones y para el envío de datos se consulta si se encuentra disponible el registro de envío (TRMT), esta situación se da cuando el PC capturardor ha leido los datos que le hemos enviado anteriormente.

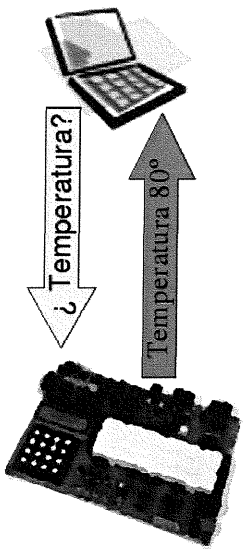

Ilustración 36: Transición cliente servidor

Se ha configurado el microcontrolador para que cuando reciba un carácter, produzca la interrupción de recepción RCI. La rutina de atención a la interrupción guarda el valor del carácter recibido y actualiza el valor de la variable updateRecibido a cierto (Nuevo carácter recibido).

Cuando el programa principal da paso al proceso de transmisiones USART, se comprueba si hay caracteres pendientes de transmitir en el vector buffTx. Si los hay y el registro de transmisión está libre, se envía el siguiente bit del vector.

Seguidamente se comprueba si se ha recibido un nuevo carácter, petición. Si se ha recibido, se carga en el vector de transmisión (buffTx) el mensaje a transmitir, en respuesta a la petición del cliente. El mensaje se inicia con el carácter "#" y se termina con el carácter "!". El primer carácter de la información transmitida corresponde al carácter de la petición, seguido del valor asignado al registro solicitado. Por ejemplo, el PC capturador envía la petición "T", en este caso el simulador responderá con el mensaje "#T80!" que significa que la temperatura es de 80ºC.

4.7.3.6 Procesar escrituras en pantalla

Con el fin de ofrecer información sobre estado de la máquina en el propio sitio de trabajo, se emplea la pantalla LCD que incorpora el PIC'SCHOLL. Existen 5 menús diferentes que muestran el valor actualizado de un sensor en cuestión.

Ilustración 37: Pantalla LCD

La pantalla dispone de 11 entradas, 3 bits de control y 8 bits de datos que se emplean para introducir comandos. Los comandos de la pantalla y su uso, se describen en el Anexo 1 del Manual de usuario del PIC'SCHOLL, también aparece el tiempo de ejecución de cada uno. Los comandos se emplean para borrar todo contenido de la pantalla, desplazar el cursor, configurar las propiedades del cursor, escribir caracteres en pantalla, etc.

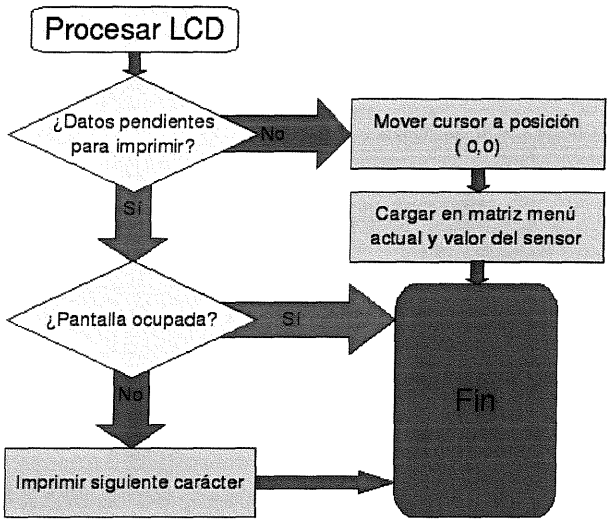

Ilustración 38: Procesar LCD

Para facilitar el trabajo con la pantalla se ha implementado la librería lcd_cxx con un juego de instrucciones básicas, como por ejemplo, inicializar la pantalla con la configuración estándar, introducir un carácter, borrar pantalla, etc.

La librería control_lcd decide que menú mostrar y cuando, consumiendo el mínimo número de recursos. Dispone de la matriz lcdscreen, que contiene los caracteres a mostrar en la siguiente impresión de pantalla, y de dos variables que apuntan a la fila y la columna del siguiente carácter

a imprimir.

El funcionamiento de la rutina consiste en comprobar si hay caracteres pendientes de imprimir en la matriz, si la pantalla no esta ocupada atendiendo un comando anterior, en caso afirmativo se imprime el carácter apuntado por el cursor. Si esta ocupada se termina la rutina de proceso de la pantalla. Si no hay nada pendiente de impresión se mueve el cursor de escritura a la posición (0, 0) y se carga en la matriz el siguiente menú a imprimir con el valor actualizado del sensor, por ejemplo si está mostrando el valor de un sensor se actualiza la pantalla con el valor del sensor en el instante de cargar los datos en la matriz.

4.7.3.7 Procesar estado de la máquina

En este punto se lee el valor de la entrada correspondiente al interruptor de estado de la máquina y se actualiza la variable simstatus. Para este pulsador no se tiene en cuenta el efecto rebote porque no se cuenta cuantas veces se ha pulsado el interruptor, porque se refresca en cada vuelta del bucle y porque se ahorran recursos del microcontrolador.

4.7.3.8 Contar eventos exteriores

Se ha de contabilizar el número de eventos que se reciben a través de la entrada T1CKI, estos eventos llegan en forma de pulso digital[5]. En el caso del simulador se simula el control de la velocidad de rotación de un motor, el cual dispone de un sensor que produce un pulso en cada rotación del eje del motor.

Es necesario programar el Timer1 para realizar este trabajo, por tanto, lo programamos de forma que cada vez que reciba un flanco de subida genere una interrupción. La rutina de atención a la interrupción incrementa el número de eventos recibidos y prepara el timer1 para que capte el siguiente evento.

Cada 15 segundos se actualiza la velocidad media del motor y se reinicia la variable que cuenta los eventos transcurridos. Se comprueba si han transcurrido los 15 segundos en la rutina de atención a las interrupciones del Timer0, mediante la operación segundos%15, que simplifica la comprobación en comparación a otros métodos.

5 Señal digital que comienza con un flanco de subida y termina con un flanco de bajada.

5 Terminal local de captura

El terminal de captura es un PC que se encuentra en el entorno de trabajo de la máquina que se ha de controlar, realiza dos funciones, la captación y transmisión de los valores de los sensores instalados en la máquina y la captura y procesado de vídeo.

Se ha empleado un PC portátil para esta tarea aunque cualquier PC que disponga de puertos USB y soporte el procesamiento de imágenes, es suficiente. El PC se encuentra conectado al simulador mediante un cable serie, que dispone de un adaptador USB serie para poder conectarlo sin problemas al portátil. También dispone de una conexión Ethernet conectada a la red del servidor central para intercambiar información.

El sistema operativo instalado en el equipo es Linux y la distribución Ubuntu 8.04., se ha escogido este sistema porque es libre, no supone ningún coste, facilita la programación de dispositivos y buses. Se ha escogido esta distribución por su amplia compatibilidad con dispositivos y la amplia comunidad de usuarios.

5.1 Captura del estado de los sensores y publicación

La captura del estado de los sensores y la publicación se realiza mediante un programa escrito en C y funciones Linux, el programa recoge el valor de los sensores del simulador y los comparte mediante el protocolo TCP. El programa es cliente y servidor, se comunica con el Simulador como cliente, consulta el valor de los sensores, trata el valor y lo transforma a las unidades entendibles por el usuario. Por último, ejerce la función de servidor compartiendo el valor de los sensores una vez han sido tratados.

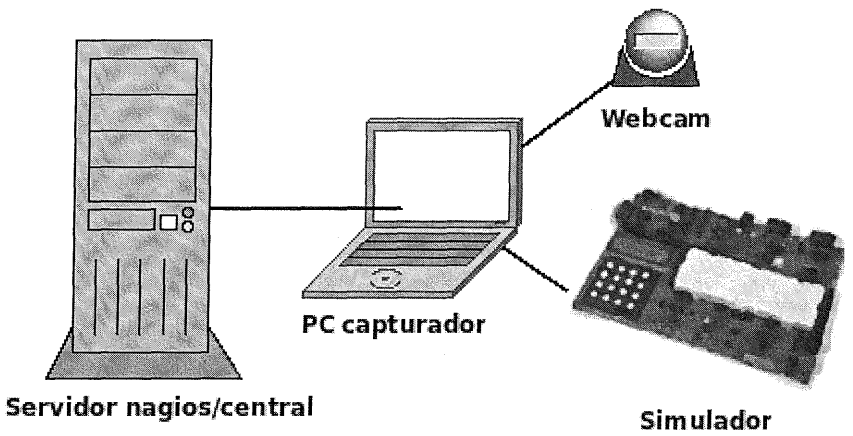

Ilustración 39: PC capturador y entorno

5.1.1 El programa capturador y servidor

El primer paso a seguir para empezar a servir información consiste en preparar el entorno. Se ha de habilitar el signal[6] SIGALRM, se ha de preparar el puerto serie para establecer la comunicación USART con el simulador y se ha de abrir un socket en modo listen (Escucha) para compartir la información con el servidor Nagios.

El programa capturador solicita al simulador el valor de los sensores de uno en uno (Petición - respuesta). Se programa una sigalrm antes de cada lectura del puerto serie para que en el caso de no recibir respuesta del simulador en 1 segundo, el programa continúe el flujo normal y evitar que quede bloqueado.

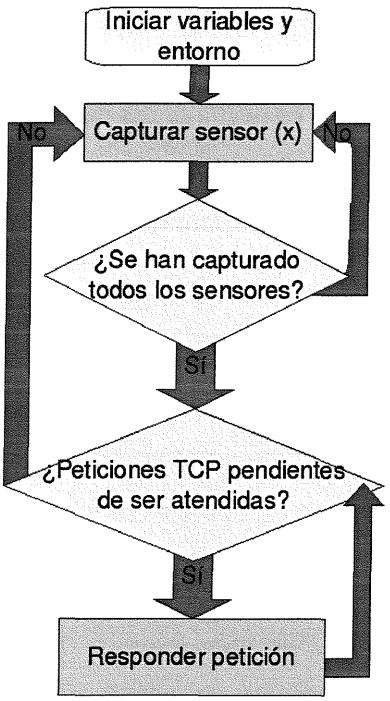

Ilustración 40: Diagrama del capturador

Cuando se han capturado todos los sensores se atienden las conexiones TCP abiertas. Se lee el socket para saber cual es el valor que solicitan y se les responde.

Al iniciar el programa es necesario pasarle como parámetros la ubicación del dispositivo controlador del puerto serie y el puerto TCP que se abrirá.

5.2 Detección de movimientos por vídeo

Para capturar imágenes se emplea la propia webcam incorporada en el portátil. La detección de movimientos se realiza mediante un software que guarda imágenes cuando se detecta movimientos en el campo de visión de la cámara y un Shell Script[7]. La tarea que desempeña el Shell Script es la de contar el número de imágenes (Ficheros) capturadas en los últimos 5 minutos.

6 Un signal (Interrupción Software) es una señal que envía el Sistema operativo a un proceso, esta señal detiene el flujo normal del proceso y pasa a ejecutar código especifico para atender la interrupción.

7 Un Shell Script es una secuencia de comandos shell que forman programa.

5.2.1 Instalación de una webcam en un sistema Linux

Los drivers que distribuyen los fabricantes no suelen incluir versiones de instalación para Linux, no obstante en internet existe una gran cantidad de drivers estándar y no oficiales que posibilitan el uso de los dispositivos de vídeo en Linux.

El primer paso para la instalación de un dispositivo de vídeo en Linux, consiste en conocer los detalles del dispositivo. Se trata de una webcam integrada en el propio chasis del portátil. Para localizarla se ejecuta el comando "lspci", que muestra todos los dispositivos PCI. En el caso del portátil no aparece en la lista, por tanto no se encuentra conectada al bus PCI del portátil.

Se continua la búsqueda con el comando "lsusb" que muestra todos los dispositivos USB. En este caso sí aparece en la lista el nombre del fabricante "OmniVision Technologies, Inc.". Para ver más detalles ejecutamos el comando "sudo lsusb -v", y obtenemos más información del dispositivo, algunos de los campos son los siguiente:

- idVendor 0x05a9 OmniVision Technologies, Inc.
- MaxPower 500mA
  Iproduct 2 Laptop Integrated Webcam
- bFunctionSubClass 3 Video Interface Collection

Con los datos obtenidos se inicia la busqueda en internet información sobre como trabajar con esta cámara en Linux y se localiza que puede trabajar instalando el driver ov51x JPEG. El driver ov51x JPEG permite trabajar con una amplia gama de cámaras, que trabajan con drivers OV, la lista de cámaras compatibles con el driver es la siguiente:

http://alpha.dyndns.org/ov511/cameras.html

Se procede a la instalación en el sistema siguiendo los pasos indicados en la weboficial del driver: http://www.rastageeks.org/ov51x-jpeg/index.php/Ov51xJpegHackedInstall#Install_debian_module_package

Finalmente, se comprueba el correcto funcionamiento con el programa camorama.

5.2.2 Motion: Software de captura y detección de movimientos

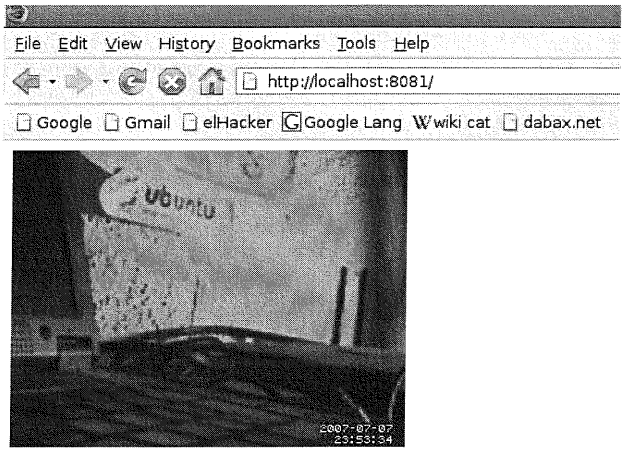

Ilustración 41: Webcam publicada mediante http

Motion es una aplicación que dispone de varias funcionalidades para trabajar con vídeo: La captura, el procesado y la publicación de vídeo.

El programa captura las imágenes directamente del dispositivo que le indicamos en la configuración "videodevice /dev/video0" y las publica en formato http, de forma que podemos controlar lo que ocurre en el campo de visión de la cámara en directo desde cualquier equipo con un navegador (PC, teléfonos móviles, PDA, etc).

Otra de las funcionalidades que implementa es la captura de imágenes cuando detecta un determinado cambio entre la última imagen capturada y la siguiente. Mediante las variables "threshold 7500" y "noise_level 94" se puede configurar el nivel a partir del cual se considera que ha habido un cambio entre una imagen y la siguiente. Threshold representa el número de píxeles que tienen que diferir en comparación a la anterior captura para que se considere que ha habido movimiento en la imagen. Noise_level es la cantidad de variación entre píxeles, es decir, si un píxel supera este valor en relación al píxel que ocupaba su posición en la anterior imagen será contado en el threshold.

5.2.3 Shell Script de consulta check_video

Con la finalidad de determinar si el número de movimientos detectados en la zona es anormal se ha creado el siguiente Script.

```
#!/bin/bash
RES=$(find /tmp/motion -type f -mmin -5|wc -l)
if [ $RES -gt 50 ]; then
echo "Detectados muchos movimientos en la zona."
exit 2
fi
if [ $RES -gt 10 ]; then
echo "Detectados movimientos en la zona."
exit 1
```

```
else
echo "No se aprecian movimientos en la zona."
exit 0
fi
```

Con el comando **find** buscamos los archivos con fecha de creación inferior a 5 minutos y con **wc** contamos el número de ficheros encontrados. Si el número de movimientos es superior a 50 imágenes se devuelve el valor de Critical "exit 2". Si es superior a 10 y inferior a 50 devuelve el valor de Warning "exit 1", si es inferior a 10 devuelve OK "exit 0".

5.3 Nagios remote plugin executor (NRPE)

El Nagios remote plugin executor es un aplicación que permite a Nagios ejecutar comandos en máquinas remotas, mediante identificación SSL[8]. NRPE actúa como servidor de consultas, se instala en la máquina a monitorizar, NRPE esta disponible para Windows y Linux. La máquina de monitorización Nagios realiza una petición, por ejemplo, check_disk (Comprobar el espacio libre en disco) y el equipo remoto responde indicando el espacio de disco libre.

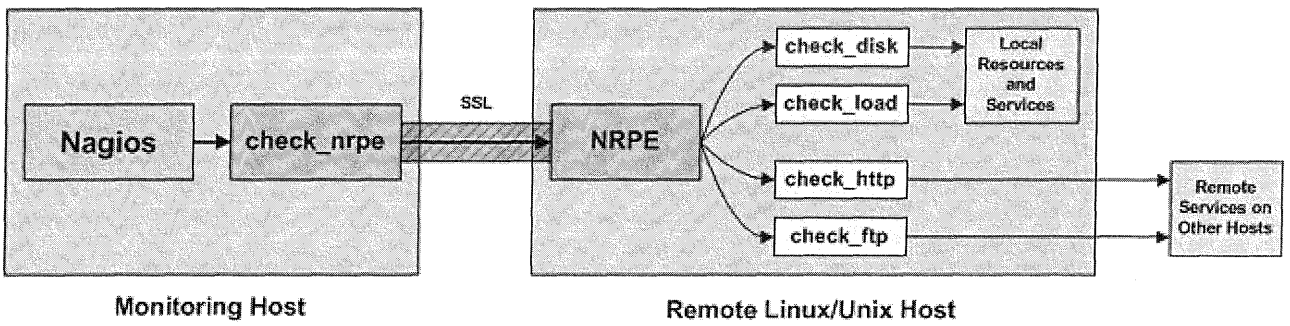

Ilustración 42: Esquema del funcionamiento de NRPE

Para poder ejecutar comandos en el PC capturador primero hay que definir los comandos en la configuración del NRPE, el ejemplo siguiente corresponde a la configuración del comando check_video que ejecuta el script check_video.

```
command[check_video]=/usr/local/nrpe-2.8/libexec/check_video
```

8 Protocolo para conexiones seguras con encriptación.

6 Sistema de monitorización

Un sistema de monitorización es un conjunto de aplicaciones o utilidades que permiten conocer el estado de equipos y servicios. Procesan los datos y disponen de una amplia variedad de funciones, estadísticas de rendimiento, gráficos, logs de estado, servicios de alerta, etc.

Antes de empezar la selección del software es necesario establecer los requisitos y funcionalidades que se han de implementar. Se han de recoger datos del simulador, procesarlos y según el estado de estos enviar alertas por SMS y correo electrónico. Ha de permitir ver el estado del sistema desde cualquier lugar.

Se pretende crear un sistema adaptable a las necesidades de cada sistema de producción y cliente, para ello se ha diseñado una maqueta que dispone de los principales elementos y más representativos que incorpora el sistema.

El esquema de la maqueta del sistema es el siguiente:

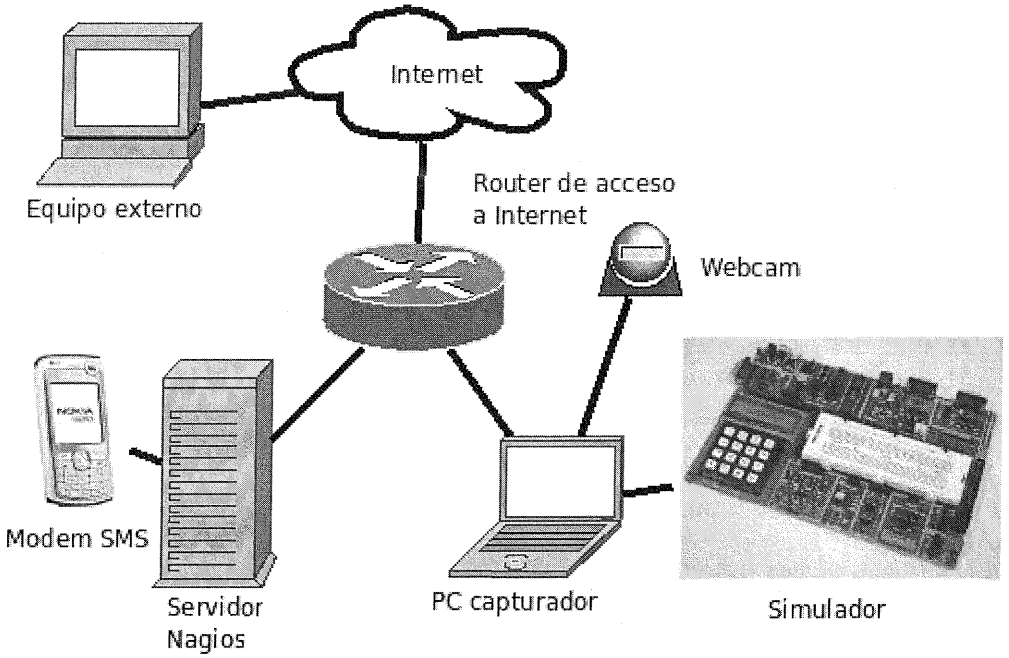

Ilustración 43: Esquema de la maqueta

El servidor Nagios realiza consultas sobre el PC capturador para actualizar el estado de los servicios y sensores. En caso de alerta se envía al usuario correspondiente un correo mediante el servidor de correo interno instalado en el Servidor Nagios, también envía un SMS mediante el teléfono móvil conectado. Para poder acceder a los servicios de la red de monitorización, los equipos externos han de establecer el túnel VPN con el Servidor Nagios. Pueden descargar su correo del Servidor nagios, consultar por web el estado de los servicios, visualizar mediante la webcam la zona monitorizada y configurar el sistema.

6.1 Sistema de monitorización Nagios

Nagios es un sistema Opensource que permite monitorizar redes. Monitoriza los host y servicios, alertando cuando el comportamiento de la red no es el deseado y cuando vuelve a su estado correcto. Nagios esta desarrollado para trabajar en Linux y algunas variantes de Unix.

Ilustración 44: Logotipo de Nagios

6.1.1 Descripción

- Permite monitorizar servicios de red (SMTP, POP, HTTP, IMAP, BD).
- Monitorización de los recursos de hosts remotos a través de tunel SSL (carga del procesador, uso de los discos, logs del sistema, procesos en ejecución, usuarios conectados, etc) mediante el Plugin NRPE disponiblem para varios sistemas operativos, linux, unix, windows y MAC OS.
- Permite la ejecución de cualquier comando de shell, con lo que el usuario puede utilizar su herramienta preferida desarrollar los comandos (Bash, C, Perl, PHP, ruby, etc).
- Notificaciones a los contactos cuando ocurren problemas en servicios o hosts y cuando quedan resueltos (Vía email, SMS, Jabber o cualquier otro método definido por el usuario siempre que venga acompañado de su correspondiente complemento).
- Permite definir manejadores de eventos proactivos que al ocurrir un suceso predefinido ejecutan acciones en un servicio o host para intentar subsanar la situación o solucionar la incidencia.
- Interfaz web opcional, que permite observar el estado de la red en el momento, notificaciones, historial de problemas, archivos de registros, etc.
- Dispone de una amplia gama de complementos que amplían las funcionalidades de Nagios. Funcionalidades como la creación de gráficos (PNP), análisis del impacto del rendimiento de los equipos sobre el negocio, exportación de datos a bases de datos Mysql.

6.1.2 Requisitos del sistema y licencia

Para funcionar correctamente Nagios necesita una maquina con Linux o una variante de Unix y el compilador de C. Para monitorizar equipos de red es necesario tener el protocolo TCP/IP correctamente configurado. Para disponer de la interfaz web es necesario instalar un servidor web, como Apache y las librerías GD en las versión 1.6.1 o superior.

Nagios se encuentra bajo licencia GNU General Public License, por lo que es posible legalmente copiarlo, distribuirlo y modificarlo bajo una ciertas condiciones. Se puede encontrar

más información acerca de esta licencia en http://www.gnu.org/copyleft/gpl.html.

6.1.2 Archivos de configuración de Nagios

La configuración de Nagios esta formada por varios archivos, localizados en la carpeta "etc" del directorio donde esta instalado Nagios por defecto *"/usr/local/nagios/etc/"*. Existen aplicaciones web que ayudan a la configuración de nagios, pero puede resultar más efectivo y ofrece más control sobre el programa la modificación los archivos plantilla que vienen por defecto. Hay cuatro tipos de archivos de configuración Main Configuration File, Resource File(s), Object Definition Files y CGI Configuration File; la definición de las variables de configuración de los archivos se encuentra el apartado Configuring Nagios, del manual de Nagios3.

6.1.2.1 *Main Configuration File*

Se trata del archivo que define el funcionamiento del demonio[9] de Nagios. El archivo se encuentra normalmente localizado en /usr/local/nagios/etc/nagios.cfg, en el se ha indicado donde se encuentra el archivo de logs del demonio, archivos y directorios de objetos, el archivo de recursos, el de cgi, el usuario con el que se ejecuta el demonio, en él se definen límites, tiempos de espera (timeout) y otras opciones menos relevantes.

También es el encargado de incluir el resto de archivos de configuración.

6.1.2.2 Resource File(s)

Es el conjunto de archivos que puede utilizar el usuario para definir macros[10]. Se puede emplear para definir propiedades comunes para varios servicios monitorizados.

6.1.2.3 Object Definition Files

Grupo de archivos donde se encuentran definidos los objetos, que pueden ser máquinas (Hosts), servicios, ontactos, comandos, etc. En estos archivos es donde se define todo lo que se ha de monitorizar del sistema, a quien avisar, cuando y mediante que medio. Se pueden emplear tantos archivos como se crea conveniente. A continuación se explican los diferentes tipos de objectos:

9 Demonio o servicio, se denomina así a un proceso informático que se ejecuta en segundo plano en vez de ser controlado por el usuario. No dispone de interfaz directa con el usuario emplea fichero de logs.

10 Serie de instrucciones que se almacenan para que se puedan ejecutar de forma secuencial mediante una llamada.

6.1.2.3.1 Plantillas

Con el fin de no tener que definir para cada host y servicio las políticas, horas de aviso, contactos, medio de notificación, comando de chequeo de estado, intervalo de chequeo, etc; se emplean plantillas (templates). Las ventajas que ofrecen es no tener que definir para cada servicio propiedades que todos van a tener en común, por tanto, proporcionan un ahorro de tiempo y facilitan la configuración.

```
Define host{
      name pfc-servers ; Nombre del template.
      use generic-host ; Template incluido en este.
      check_period 24x7 ; Periodo de tiempo durante el que se chequea.
      check_interval 2 ; Espacio de tiempo entre chequeos, minutos.
      retry_interval 1 ; Minutos de espera para volver a chequea un
objeto en caso de no encontrarse en estado OK.
      max_check_attempts 2 ; Numero de intentos de chequeo antes de
notificar problema en un servicio.
      check_command check-host-alive ; Comando empleado para el chequeo.
      notification_period workhours ; Periodo de tiempo durante el que se
envían notificaciones.
      notification_interval 120 ; Cada cuanto tiempo se envía un
recordatorio.
      notification_options d,u,r ; Para que estados se envía el
recodatorio, ( d:down, u:unreachable, r:servicio recuperado).
      contact_groups admins ; Contactos a los que enviar las
notificaciones.
      register 0 ; Se emplea para indicar que esta definición es una
plantilla y no de un host.
}
```

6.1.2.3.2 Hosts

Se tratada de la máquina que es objeto de monitorización, los atributos típicos a definir son el tipo de máquina monitorizada (Servidor, estación de trabajo, router, switch, impresora, etc), la dirección IP de la máquina y además permite establecer relaciones entre hosts.

```
define host{ ; Definición del simulador
      use pfc-servers ; Nombre de la plantilla a la que pertenece.
      host_name simulador ; Nombre del host.
      alias Simulador ; Nombre corto del host.
      address 192.168.1.4 ; Dirección IP.
      check_command check_simulador!192.168.1.4!8080!P ;Comando que
chequea el estado del host.
}
```

En el apartado de hosts detail de la interfaz web de Nagios podemos observar el estado de todos los host monitorizados.

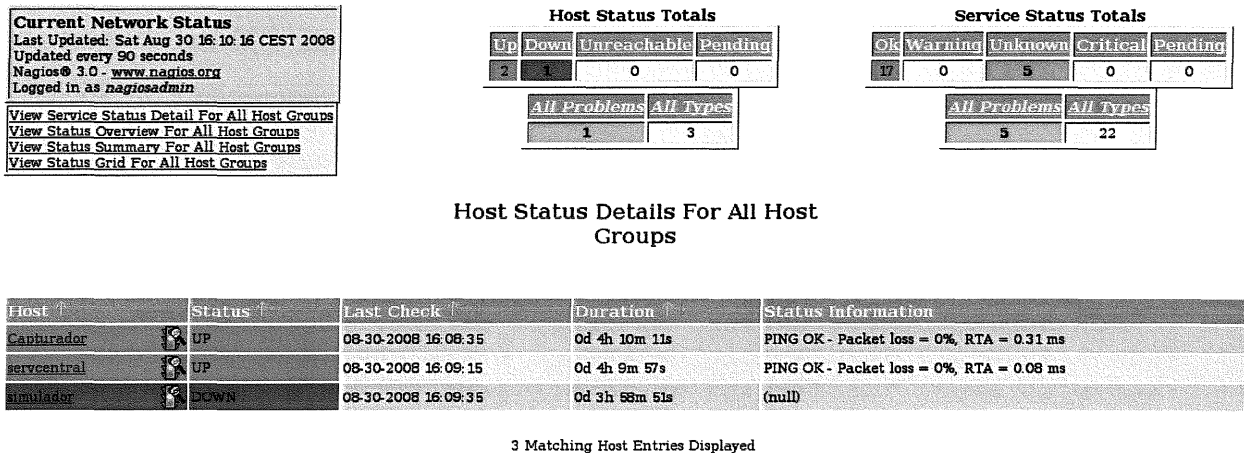

Ilustración 45: Nagios -> Host detail

En este proyecto se monitorizan tres máquinas el simulador (Maqueta de simulación de una máquina de produción), Capturador (Terminal local de captura) y el servcentral (Servidor central de monitorización).

Los host se pueden encontrar en 4 estados diferentes UP (Levantado), DOWN (Caído), UNREACHABLE (No se puede llegar hasta el host porque alguna máquina intermedia esta caída), PENDING (No se ha llevado a cabo el chequeo).

6.1.2.3.3 Host Group

Se puede emplear para crear grupos de máquinas y facilitar la configuración de los objetos.

```
define hostgroup{ ; Definición del grupo de máquinas servidor proyecto.
      hostgroup_name pfc-servers ; Nombre del grupo.
      alias PFC Lan servidores ; Nombre que aparecerá en la descripción .
      members servcentral,simulador ; Host pertenecientes al grupo.
}
```

6.1.2.3.4 Service

El concepto de Service referencia a los servicios que corren en una máquina como SMTP, POP, HTTP, etc; y valores de los elementos de las máquinas, espacio utilizado en disco, temperatura CPU, sensores, etc. En el caso del equipo simulador para cada sensor de la máquina monitorizada existe un servicio que indica, según los parámetros definidos, en que estado se encuentra cada sensor.

```
define service{
      use local-service ; Nombre del template empleado.
      host_name simulador ; Host asociado al servicio.
      service_description Temperatura del Simulador ; Descripción del
servicio.
      check_command check_simulador!192.168.1.4!8080!T ; Comando que
devuelve el estado del servicio.
}
```

En el ejemplo anterior se puede observar la declaración del servicio "temperatura de la máquina".

Service Status Details For All Hosts

Host	Service	Status	Last Check	Duration	Attempt	Status Information
Capturador	Current Load	OK	09-01-2008 16:25:05	0d 0h 8m 0s	1/4	OK - load average: 0.36, 0.37, 0.31
	Current Users	OK	09-01-2008 16:25:46	0d 0h 7m 19s	1/4	USERS OK - 4 users currently logged in
	Dector video	OK	09-01-2008 16:23:27	0d 0h 4m 38s	1/4	No se aprecian movimientos en la zona.
	PING	OK	09-01-2008 16:23:08	0d 0h 4m 57s	1/4	PING OK - Packet loss = 0%, RTA = 1.08 ms
	Root Partition	OK	09-01-2008 16:27:49	0d 0h 5m 16s	1/4	DISK OK - free space: / 79214 MB (75% inode=97%):
	SSH	OK	09-01-2008 16:24:30	0d 0h 8m 35s	1/4	SSH OK - OpenSSH_4.7p1 Debian-8ubuntu1.2 (protocol 2.0)
	Swap Usage	OK	09-01-2008 16:24:11	0d 0h 8m 54s	1/4	SWAP OK - 100% free (972 MB out of 972 MB)
	Total Processes	OK	09-01-2008 16:24:52	0d 0h 8m 13s	1/4	PROCS OK: 138 processes
servcentral	ConexionInternet	OK	09-01-2008 16:25:45	0d 0h 7m 20s	1/4	PING OK - Packet loss = 0%, RTA = 93.34 ms
	Current Load	OK	09-01-2008 16:26:00	2d 4h 27m 5s	1/4	OK - load average: 0.02, 0.01, 0.00
	Current Users	OK	09-01-2008 16:26:41	2d 4h 26m 24s	1/4	USERS OK - 1 users currently logged in
	HTTP	OK	09-01-2008 16:27:22	2d 4h 25m 43s	1/4	HTTP OK - HTTP/1.1 302 Found - 0.001 second response time
	PING	OK	09-01-2008 16:23:03	2d 4h 25m 2s	1/4	PING OK - Packet loss = 0%, RTA = 0.08 ms
	Root Partition	OK	09-01-2008 16:23:44	2d 4h 24m 21s	1/4	DISK OK - free space: / 3450 MB (77% inode=92%):
	SSH	OK	09-01-2008 16:24:25	2d 4h 23m 40s	1/4	SSH OK - OpenSSH_4.3p2 Debian-9 (protocol 2.0)
	Swap Usage	OK	09-01-2008 16:25:05	2d 4h 23m 0s	1/4	SWAP OK - 100% free (478 MB out of 478 MB)
	Total Processes	OK	09-01-2008 16:25:33	2d 4h 27m 32s	1/4	PROCS OK: 32 processes with STATE = RSZDT
simulador	Estado de la máquina	OK	09-01-2008 16:26:14	0d 0h 6m 51s	1/4	La maquina se encuentra trabajando
	Nivel del deposito	OK	09-01-2008 16:23:05	0d 0h 5m 0s	1/4	Capacidad del deposito correcta: 23
	Pulsador de emergencia	OK	09-01-2008 16:27:35	0d 0h 5m 30s	1/4	Pulsador de emergencia no pulsado.
	Temperatura del Simulador	OK	09-01-2008 16:23:16	0d 0h 4m 49s	1/4	Temperatura correcta : 102 C
	Velocidad del Motor	OK	09-01-2008 16:23:57	0d 0h 9m 8s	1/4	Velocidad de trabajo correcta: 16 RPM

Ilustración 46: Nagios -> Service detail

La Ilustración 46 es una captura de la interfaz web de nagios, en concreto del apartado Service detail, en ella aparecen todos los servicios monitorizados, se puede observar que todos ellos están funcionando correctamente.

6.1.2.3.5 Service Group

Se pueden emplear para crear grupos de servicio, por ejemplo se puede crear un grupo con todos los servicios que trabajan sobre las bases de datos.

```
define servicegroup{
      servicegroup_name dbservices ; Nombre corto del grupo.
      alias Database Services ; Nombre del grupo.
      members ms1,SQL Server,ms1,SQL Server Agent,ms1,SQL DTC ; Servicios
que pertenecen al grupo.
}
```

6.1.2.3.6 Contact

Se emplea para definir las personas de contacto, entre todas las opciones disponibles destacan las direcciones de correo, el nombre del contacto, las horas de contacto, en que casos contactar, mediante que métodos, comandos contactar, etc.

```
define contact{
      contact_name ricardofrias ; Nobre corto del contacto.
      use generic-contact ; Template empleado.
      alias Ricardo Frías Alvarez ; Nombre completo del contacto.
      email guardias ; Dirección de correo.
      service_notification_period 24x7 ; Periodo de notificación de
sucesos en servicios
      host_notification_period 24x7 ; Periodo de notificación de sucesos
en hosts
      service_notification_options c ; Notificar cuando un servicio pase
a estado crítico
      host_notification_options d ; Notificar cuando un host pase a
estado down.
      service_notification_commands notify-service-by-sms!6xxxx ; Comando
empleado para notificar problema en un servicio.
      host_notification_commands notify-host-by-sms!67xxxx ; Comando
empleado para notificar problema en un host.
}
```

6.1.2.3.7 Contact group

Este tipo de objeto se emplea para definir grupos de contactos, lo que permite avisar a varias personas con un único comando.

```
define contactgroup{
      contactgroup_name admins ; Nombre corto del grupo
      alias Nagios Administrators ; Nombre completo.
      Members ricardofrias, Guardias ; Miembros del grupo.
}
```

6.1.2.3.8 Time Period

Se emplean para definir periodos de tiempo que después se pueden emplear para enviar avisos durante estos periodos predefinidos o chequear durante los mismos.

```
define timeperiod{
      timeperiod_name misc-skip-ranges
      alias Misc Skip Ranges
      2007-01-01 - 2008-02-01 / 3 00:00-24:00 ; Every 3 days from January
1st, 2007 to February 1st, 2008
      2008-04-01 / 7 00:00-24:00 ; Every 7 days from April 1st, 2008
(continuing forever)
      monday 3 - thursday 4 / 2 00:00-24:00 ; Every other day from 3rd
Monday to 4th Thursday of every month
      day 1 - 15 / 5 00:00-24:00 ; Every 5 days from the 1st to the 15th
day of every month
      july 10 - 15 / 2 00:00-24:00 ; Every other day from July 10th to
July 15th of every year
      tuesday 1 april - friday 2 may / 6 00:00-24:00 ; Every 6 days from
the 1st Tuesday in April to the 2nd Friday in May of every year
}
```

6.1.2.3.9 Command

Los comandos son funciones que se definen en la configuración de nagios y más adelante pueden ser llamados, pueden ejecutar programas, scripts y admiten parámetros. Pueden ser llamados para enviar notificaciones, chequear servicios, hosts, para solucionar problemas, etc.

```
define command{
      command_name check_simulador ; Nombre del comando.
      command_line /usr/local/bin/consultas-nagios $ARG1$ $ARG2$ $ARG3$ ;
Comando a ejecutar.
}
```

Un ejemplo de llamada al comando anterior sería el siguiente: check_command check_simulador!192.168.1.4!8080!T. el primer argumento es el nombre del comando (check_simulador) y el resto de argumentos están separados por el simbolo "!". Para pasar los parámetros al programa que ejecuta el comando se realiza mediante la etiqueta $ARGx$, donde x es el número de argumento (0 corresponde al nombre del comando).

6.1.2.4 CGI Configuration File

Archivo donde se define el lugar en el que se encuentran localizados los diferentes archivos y los permisos de los usuarios para el sistema de la interfaz web (Archivos de configuración, de web, directorio raíz, etc).

6.1.3 La configuración

Se ha configurado Nagios para monitorizar 3 máquinas y un total de 22 servicios, Servidor Nagios(8 servicios), PC capturador (9 servicios) y simulador (5 servicios). Cada host tiene un archivo de configuración asociado, donde se define el host y los servicios asociados, los archivos son localhost.cfg, sensores.cfg y simulador.cfg, respectivamente.

Para realizar la configuración se han aprovechado los archivos de configuración de muestra que aparecen con la instalación. Se han aprovechado los comandos que vienen predefinidos para la monitorización de sistemas linux y se han creado nuevos comandos, para chequeos específicos de este proyecto y envío de SMS.

Los chequeos más destacables teniendo en cuenta la naturaleza del proyecto son los que se realizan sobre el Simulador, "Estado de la máquina", "Nivel del depósito", "Pulsador de emergencia", "Temperatura del Simulador" y "Velocidad del Motor", el chequeo de los sensores del simulador se lleva a cabo mediante un programa cliente que contacta con el PC capturador, obtiene las mediciones y determina el estado del servicio.

Fuera del ámbito del simulador cabe destacar los chequeos de la conexión a internet del servidor realizando ping a sitios web externos y el chequeo de movimientos en la zona de vídeo ya comentado anteriormente. El resto de servicios son estándar y se realizan sobre todas las máquinas Linux.

6.1.4 Como funciona

Sobre cada servicio y host Nagios asigna una fecha y hora para realizar el próximo chequeo. Cuando llega el momento, se ejecuta el comando definido para chequear el servicio. Nagios recoge el estado actual (Current Status) del servicio a partir del valor de terminación del proceso(comando) que ejecutó "exit (int status);" y recoge la información del estado (Status Information) a partir del valor devuelto por el comando a través de la salida estándar. Los estados devueltos pueden ser 4: OK, WARNING, CRITICAL o UNKNOWN.

Service State Information

Current Status:	OK (for 0d 12h 57m 55s)
Status Information:	PING OK - Packet loss = 0%, RTA = 0.99 ms
Performance Data:	
Current Attempt:	1/4 (HARD state)
Last Check Time:	09-02-2008 17:36:11
Check Type:	ACTIVE
Check Latency / Duration:	0.005 / 4.022 seconds
Next Scheduled Check:	09-02-2008 17:41:11
Last State Change:	09-02-2008 04:41:11
Last Notification:	N/A (notification 0)
Is This Service Flapping?	NO (0.00% state change)
In Scheduled Downtime?	NO
Last Update:	09-02-2008 17:38:58 (0d 0h 0m 8s ago)

Ilustración 47: Información de estado

Si el sistema detecta un cambio en el estado del servicio vuelve a chequearlo para confirmar el cambio y acto seguido se notifica al contacto que corresponde, ejecutando el comando configurado para este fin. El aviso de error se envía periódicamente mientras perdura la incidencia (Estado diferente de OK). Por último, se programa una fecha para el siguiente chequeo del servicio o host.

Valor	Estado
0	OK
1	WARNING
2	CRITICAL
3	UNKNOWN

Ilustración 48: Estado según valor de terminación

6.2 Obtención del estado de los sensores de la máquina

El proceso de determinación del estado de los sensores comienza por la captura del valor de los sensores. El sistema implementado en este proyecto esta compuesto por tres puntos:

1. El simulador: Donde se obtiene directamente de los sensores (Hardware) la medición.
2. El PC capturador que constantemente consulta al simulador el estado de los sensores y hace de puente entre el simulador y el protocolo TCP/IP.
3. El Servidor Nagios que mediante un Cliente TCP/IP consulta al PC capturador el valor de los sensores y por último determina el estado de estos.

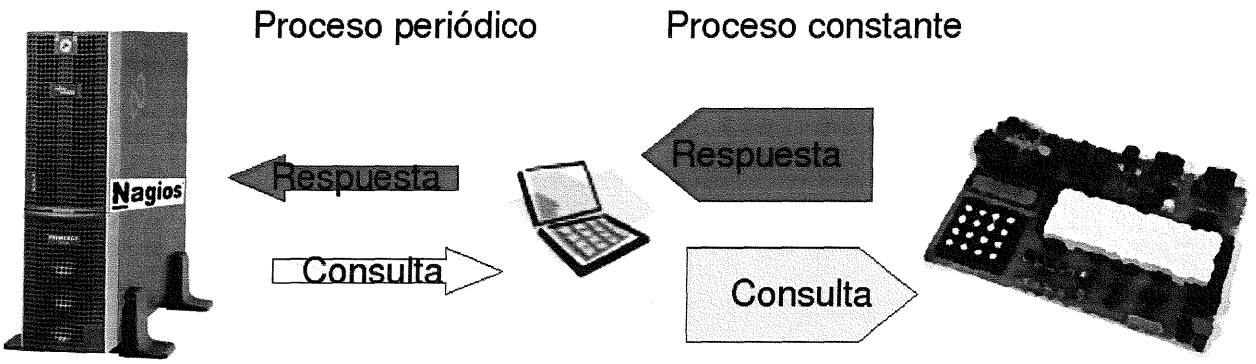

Ilustración 49: Proceso de obtención del estado de los sensores

Este apartado se centra en el proceso realizado por el Servidor Nagios, ya que el resto de puntos del sistema han sido explicados en apartados anteriores.

Se ha implementado un programa en C, llamado consultas-nagios.c. Este programa recibe tres parámetros, la IP del PC capturador al que se realiza la consulta, el puerto TCP abierto en el servidor (PC capturador) para este tipo de peticiones y el código de consulta. Con estos datos el programa se conecta al servidor instalado en el PC capturador y le envía el código de consulta. El PC capturador responde con el valor del sensor correspondiente al código de consulta. Finalmente el servidor Nagios compara el valor recibido con las constantes predefinidas, y determina el estado del sensor. Se emplea el estado de terminación del proceso "exit (int status);" para indicar el

estado del sensor consultado.

El comando de Nagios que realiza esta función es check_simulator y es llamado para consultar el estado de los sensores de todos los servicios del simulador incluido el ping, que se emplea para saber el estado de la máquina.

6.3 Alertas Correo electrónico

El sistema ha de estar preparado para mandar correos electrónicos cada vez que un servicio o host cambia de estado. Para implementar este sistema, es necesaria la instalación de un MTA[11] (Mail Transport Agent) en el servidor Nagios. Se ha procedido a la instalación de Postfix como cliente de correo. Se ha configurado como cliente de entrega local, es decir, solo puede recibir correo y no enviar a otros servidores, a continuación se detallan los motivos.

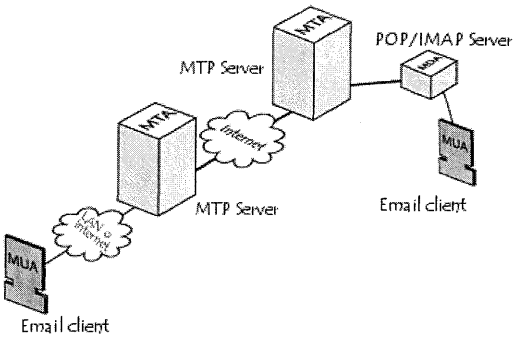

Ilustración 50: Servicio de correo

Uno de los principales problemas de la informática en la actualidad es el Correo Basura (Spam). La mayoría de servidores de internet aplican reglas para evitar la recepción de estos correos, listas negras, listas grises, algoritmos de detección, resolución inversa, comprobación de cuentas, etc. Hay una de las medidas antispam que complica que desde el servidor Nagios se envíen los correos a otros servidores de correo, requieren de un nombre de dominio existente.

No es un problema una vez el producto se vende al cliente porque existen varías soluciones al problema. Una opción es que el servidor Nagios se convierta en el servidor de correo del cliente. Otra opción consiste en enviar correos a través de un servidor de correo externo utilizando identificación. Y la última opción propuesta es añadir la IP del servidor de nagios a la lista blanca del servidor de destino.

Otro de los motivos por los cuales se ha escogido este modelo de servidor es para no depender de la conexión a internet en la maqueta.

Con el MTA el sistema recibe correo, pero es necesario acceder desde el exterior para descargar el correo. Courier Mail Server se trata de un MDA[12] que incorpora un servidor de correo POP y un servidor de correo IMAP, que nos permite leer el correo electrónico con nuestro cliente de correo y se encarga de la entrega de los correos en el buzón del usuario. Las cuentas de

11 Se trata de un programa que transfiere el correo de una máquina a otra.
12 Mail delivery agent se trata de un programa que acepta el correo y lo entrega en el buzón.

correo son los nombres de usuario del sistema; Postfix se les entrega el correo en la carpeta Maildir del directorio home de cada usuario.

Lo único que el usuario tiene que hacer para recibir avisos por correo es configurar su cuenta de usuario en su cliente de correo preferido (Thunderbird, Evolution, Outlook, etc), como IMAP o bien como POP.

6.3.1 Selección del MTA

En el proceso de selección del MTA se han tenido en cuenta aquellos MTA que son compatibles con Linux y gratuitos. Se han comparado 4, el más antiguo de todos **sendmail** desarrollado por Eric Allman, **qmail** escrito por D.J. Bernstein's, **postfix** de Wietse Venema's y **exim** desarrollado en la Universidad de Cambridge.

Sendmail es el que más historia tiene de todos, fue escrito en 1980, se caracteriza por se difícil de configurar y en consecuencia nos permite un control total sobre el correo. Durante los últimos años se han detectado muchas vulnerabilidades en su funcionamiento que progresivamente se han ido solucionando.

Ilustración 51: Sendmail logotipo

Qmail fue el primer MTA con seguridad que se publico, en aquel momento sendmail no contaba con opciones de seguridad. Qmail al contrario que Sendmail es fácil de configurar en los principales aspectos gracias a los ficheros pequeños en texto plano de configuración. Funciona de forma modular, los procesos trabajan por separado haciendo una función distinta cada uno (qmail-queue, qmail-qmtpd, qmail-send, etc).

Ilustración 52: Qmail logotipo

Las características más destacables de Postfix es la facilidad de administración, la velocidad de proceso de los mensajes y la seguridad que ofrece respecto a sendmail. Cuenta con gran número de medidas Antivirus y Antispam. TLS, Greylist, listas DNSBLs, RFC, whitelist, blacklist, etc. Además puede funcionar con diferentes bases de datos Berkeley DB, CDB, DBM, LDAP, MySQL y PostgreSQL.

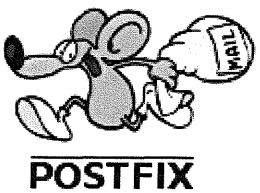

Ilustración 53: Postfix logotipo

Exim apareció en 1995 como el substituto de sendmail, es compatible con la mayoría de comandos de configuración de sendmail, pero también compartía muchas de las vulnerabilidades de sendmail. Actualmente las reglas de seguridad de Exim son comparables a las de postfix.

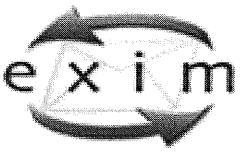

Ilustración 54: Exim logotipo

Cualquiera de los 4 MTA analizados son buenos candidatos, pero finalmente se ha escogido Postfix por las ventajas ya mencionadas, porque se puede instalar fácilmente y por la experiencia acumulada con este sistema.

6.4 Alertas SMS

El servicio de mensajes cortos o SMS (Short Messge Service) es un servicio disponible en los teléfonos móviles que permite el envío de mensajes de texto cortos, aunque actualmente ya está disponible para teléfonos fijos y otros dispositivos. Los SMS ofrecen ventajas frente a otros sistemas de alerta como por ejemplo:

- Pueden ser recibidos por cualquier teléfono móvil o otros dispositivos.
- Permiten al usuario la monitorización pasiva
- Permiten al usuario recibir avisos en cualquier lugar.

En los mensajes que envía Nagios se incluye información útil para indicar al usuario que servicio está afectado y en que grado.

Para enviar SMS existen diferentes opciones:

- Envío a través de servidores internet
- Mediante Modem GSM
- A través de correos electrónicos redireccionados a números de teléfono
- Mediante línea telefónica.

A continuación se detallan y se explica los motivos por los que se ha escogido el envío a través de modem GSM.

6.4.1 SMS a través de internet

Se trata de un servicio que ofrecen algunas empresas que permite a los PC mediante la instalación de una aplicación el envío de SMS, a través de la conexión a internet. Es muy utilizado en el entorno web para el envío de confirmaciones de reservas, envío de claves de confirmación, etc.

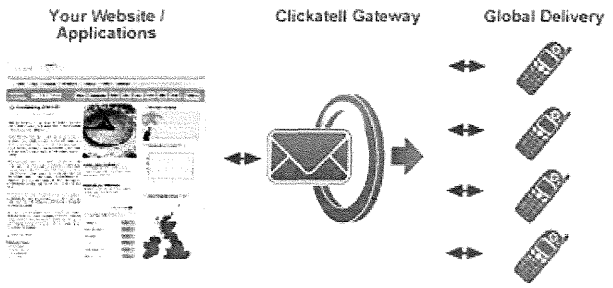

Ilustración 55: Esquema SMS a través de internet

El funcionamiento del servicio es el siguiente: la aplicación, en este caso nagios, conecta con el servidor del proveedor (Gateway), le entrega el mensaje y este se encarga de entregarlo a los teléfonos destinatarios.

El precio del servicio en comparación a los proveedores de telefonía móvil es ligeramente inferior, como se puede observar en la tabla de la Ilustración 56, se ha tenido en cuenta el peor de los casos para realizar la tabla.

Proveedor	Precio SMS (sin IVA)
esendex.es	0,14 €
arrakis.com	30,00 € (300 SMS/Mes)
mensatek.com	5 € (49 SMS)
clickatell.com	0.088 €
Orange	0.15 € / 0.05 € (A Orange)
Vodafone	0.15 € / 0.075 (A Vodafone)
Movistar	0.15 €
Yoigo	0.10 €

Ilustración 56: Tabla comparativa precios

6.4.2 Cuentas de correo redirigidas a números de teléfono

Se trata de un servicio que consiste en reenviar los emails recibidos en una cuenta de correo por SMS a un número de teléfono. Funciona de la siguiente forma, el PC de remitente entrega un correo electrónico a su servidor SMTP, este lo entrega al servidor SMTP del proveedor de telefonía móvil del destinatario. Una vez el Servidor SMTP del proveedor recibe el correo, comprueba la cuanta a la que va dirigido y si corresponde envía un mensaje a través del protocolo SMS al número de teléfono asociado a la cuenta de correo. El precio de este servicio es el mismo que el mismo que el de envío de SMS a través de internet, dependiendo el tamaño del correo se enviará un mensaje o varios.

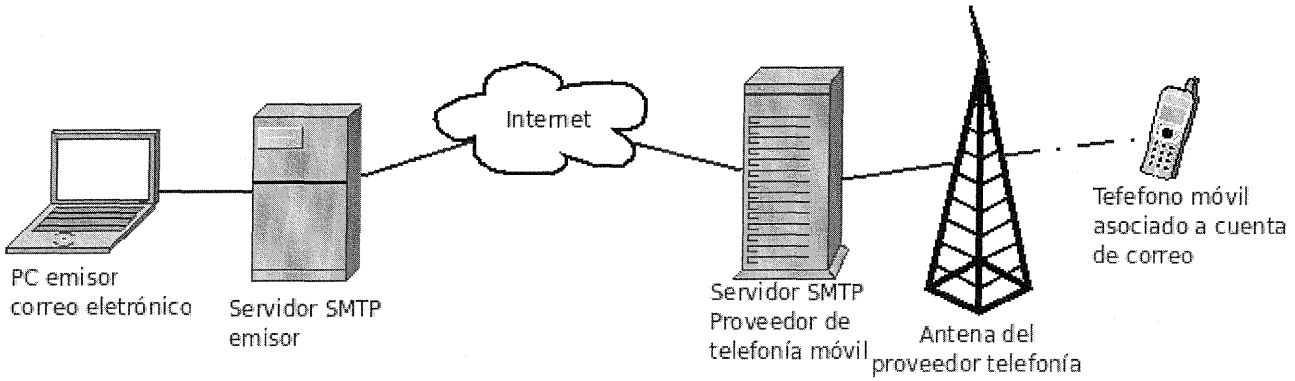

Ilustración 57: Esquema servicio redirección de email a teléfono móvil

No obstante, existe una alternativa gratuita que ofrecen algunas operadoras de telefonía, consiste en un servicio que permite crear cuentas de correo asociada a un número de teléfono. Cuando se recibe un correo en la cuenta de correo el servidor envía un aviso de recepción de correo por SMS al número asociado a la cuenta, el mensaje que contiene la dirección de correo del remitente del correo y el asunto, limitado por un número de caracteres. El destinatario una vez recibe el aviso puede acceder al correo a través del teléfono móvil o mediante un cliente de correo.

6.4.3 SMS mediante la línea telefónica

Se trata de un servicio que ofrecen las operadoras de teléfono, consiste en el envío de SMS desde líneas telefónicas (RTC y RDSI). Las operadoras ofrecen terminales en forma de teléfonos fijos e inalámbricos que permiten el envío de SMS, pero no están diseñados para ser conectados a un PC y ser emplearlos como modems SMS.

Ilustración 58: Modem SMS sobre RTC

Para implementar este servicio es necesario un modem SMS sobre línea telefónica, un buen candidato es el Z-text SMS Modem, que dispone de drivers del fabricante para Linux 2.4, 2.6, MAC OS9 y OSX. Permite tanto el envío como la recepción de mensajes, con lo que permite añadir funcionalidades como por ejemplo al recibir un determinado SMS parar la máquina. El precio aproximado del equipo es de 100€ y el precio de envío por SMS a móvil es de 0.15 € aproximadamente.

6.4.4 SMS a través GSM

Se trata del mismo protocolo de envío de SMS que emplean los teléfonos móviles, el GSM. El sistema esta formado por un equipo que envía mediante un programa los mensajes al módem GSM o teléfono móvil que tiene conectado. El módem o móvil lo envía a través de la red GSM al teléfono móvil destinatario.

Existen equipos específicos para esta función, módems GSM, pero es posible también enviar SMS a través GSM conectando un teléfono móvil que soporte comandos AT[13] de envío de SMS, esta última es la opción más económica.

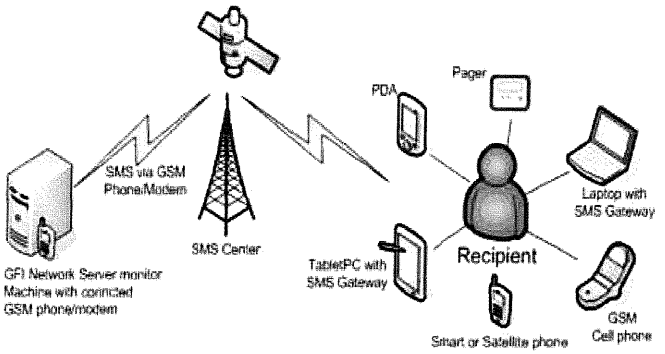

Ilustración 59: Esquema servidor SMS a través de GSM

En el proyecto se ha implementado el servicio de alertas por SMS esta tecnología teniendo en cuenta las siguientes ventajas y desventajas:

- Si cae la conexión a internet se puede informar al usuario a través de la red GSM, cosa que no es posible con los métodos de envío a través de internet y correo electrónico.

13 Los comandos AT son un grupo instrucciones que se emplean generalmente para configurar y trabajar con módems.

- El precio del envío de correo es el mismo que de móvil a móvil, dependiendo la tarifa aplicada puede ser más económico el envío a través de internet.
- Los avisos de recepción de correo (Cuentas redirigidas a teléfonos móviles) son gratuitos pero no contienen toda la información.
- En el caso del envío a través de línea de teléfono fijo, es necesario adquirir un equipo externo especifico para implementar el servicio.
- Para enviar SMS a través de las líneas RTC, es necesario hacer llegar un cable de línea hasta el módem. Con el módem GSM no es necesaria ninguna instalación solo es necesario tener cobertura.

6.4.5 Implementación de las alertas SMS

El sistema de envío de SMS de este proyecto esta compuesto por un teléfono móvil que soporta comandos AT (Nokia N70) conectado al servidor de Nagios mediante de un cable USB. Para permitir a Nagios enviar correos se ha programado un comando de Nagios y un programa escrito en C que envía SMS.

6.4.5.1 Comados AT

Los comandos AT son instrucciones codificadas que conforman un lenguaje de comunicación entre el hombre y terminal módem. El juego de comandos AT fue desarrollado en 1977 por Dennis Hayes como un interfaz de comunicación con un modem para configurar y proporcionar instrucciones a los terminales, que permiten por ejemplo marcar un número de teléfono, descolgar, etc. Los comandos AT se denominan así por la abreviatura de attention. La telefonía móvil GSM también ha adoptado como estándar este lenguaje para poder comunicarse con sus terminales (Teléfonos móviles). Para ello se han ampliado los comandos y ahora permiten acciones tales como realizar llamadas de datos o de voz, leer y escribir en la agenda de contactos y enviar mensajes SMS, además de muchas otras opciones de configuración del terminal.

Se puede acceder a un listado de los principales comandos en la siguiente páguina:

http://wiki.forum.nokia.com/index.php/AT_Commands#NOKIA_GSM_AT_COMMAND_SET

El primer paso para enviar SMS consiste en encontrar los comandos AT necesarios para enviar SMS y comprobar si el teléfono móvil soporta el envío de SMS a través de comandos AT, para ello nos conectamos al dispositivo a través de un programa de emulación de terminal y control de módem como minicom o Hyperterminal (Windows), e introducimos el comando "AT+CMGS=?". Si la respuesta es OK, el móvil permite el envío de SMS.

La secuencia de comandos para el envío de SMS es la siguiente:

```
at
[OUTPUT]OK
at+cmgf=?
[OUTPUT]+CMGF: (0,1)
[OUTPUT]
[OUTPUT]OK
at+cmgf=1
[OUTPUT]OK
at+cmgs=?
[OUTPUT]OK
at+cmgs="<telephone number to send sms>"
> <text of the sms>
> ->(end of text message: press CTRL + Z )
[OUTPUT]OK
```

6.4.5.2 Programa envío de SMS

El programa send_sms.c, implementado en C, recibe tres parámetros, ubicación del dispositivo asociado al teléfono móvil, número de teléfono al que está conectado y mensaje. Un ejemplo de llamada al programa sería el siguiente:

```
$./send_sms /dev/ttyACM0 6xx.xxx.xxx "Mensaje de prueba 1.0v"
```

El programa establece una conexión serie con el modem GSM a través del puerto indicado y mediante comandos AT envía el SMS al número indicado. La conexión en este caso se realiza a través de un cable USB, pero también es posible realizarla mediante Bluetooth.

6.4.5.3 Configurar envío de SMS en nagios

Nagios se encarga de llamar al programa de envío de SMS cuando alguna máquina del sistema se ha caído, el comando definido para esta función es notify-host-by-sms. El mensaje se envía al número pasado como argumento en la llamada (service_notification_commands notify-service-by-sms!6xxxxxxxx). El texto enviado por Nagios contiene la siguiente información: Se trata de un aviso de nagios, la máquina afectada, el estado, información adicional y la hora a la que se detectó el problema. La definición del comando es la siguiente:

```
# 'notify-host-by-sms' command definition
define command{
      command_name notify-host-by-sms
      command_line /pfc/send_sms /dev/ttyACM0 $ARG1$ "***** Nagios *****\
n\nHost: $HOSTNAME$\nState: $HOSTSTATE$\nAddress: $HOSTADDRESS$\nInfo:
$HOSTOUTPUT$\n\nDate/Time: $LONGDATETIME$\n" >> /var/log/sms.log
}
```

En caso de que se necesario también es posible enviar un SMS cuando haya algún problema con los servicios para ello hay que llamar al comando notify-service-by-sms, que envía un SMS indicando el nombre del servicio afectado, la máquina en la que se encuentra y el estado del mismo.

```
define command{
      command_name notify-service-by-sms
      command_line /pfc/send_sms /dev/ttyACM0 $ARG1$  "***** Nagios *****
Service: $SERVICEDESC$  Host: $HOSTALIAS$  State: $SERVICESTATE$"
}
```

6.5 Plataforma web

En el siguiente apartado se introduce el entorno web de Nagios y explican las funcionalidades interesantes de este programa para el proyecto. La web de nagios se encuentra alojada en el servidor central y se emplea el servicio Apache para la publicación. La web está programada en HTML, CSS y utilíza scripts CGI; lo que nos permite adaptarla y personalizarla según las necesidades del cliente. Tras analizar la herramienta se ha considerado que la plataforma web proporcionada por Nagios es suficiente y se ajusta a las necesidades de este proyecto, por lo tanto no es necesario adaptarla como se contempló en los objetivos.

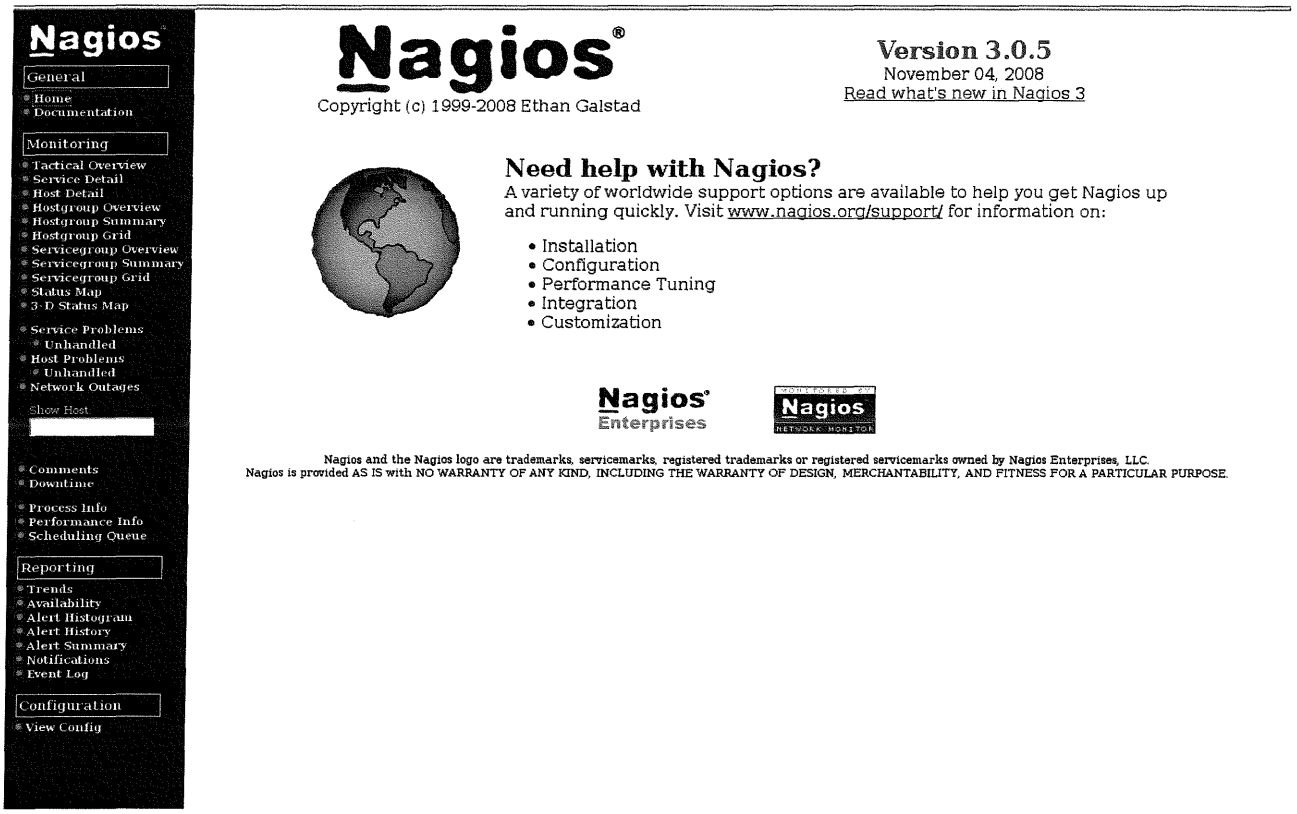

Ilustración 60: Home de la web de Nagios

En los siguientes apartados se explican las diferentes funcionalidades del menú web de Nagios.

6.5.1 Tactical Overview

El apartado Tactical Overview muestra un resumen del estado de todos los servicios y host. Los host se pueden encontrar en 4 estado diferentes.

- Up: Indica que no hay ningún problema y el host es accesible.
- Down: Se produce cuando no obtenemos respuesta del host. La situación puede ser critica.
- Unreachable: Se da esta situación cuando algún host intermedio entre el servidor Nagios y el equipo final se encuentra caído y no puede comprobar el estado del host final.
- Pending: Es el estado inicial en el que se encuentran los host cuando Nagios nunca ha comprobado su estado, pendiente de ser comprobado.

Los servicios monitorizados se pueden encontrar en 5 estados:

- Ok: Funciona de forma apropiada según los valores establecidos.
- Warning: Indica que hay una desviación ligera de los valores normales pero sin llegar a ser grave.
- Critical: Se produce cuando los valores obtenido corresponden a una afectación grave o muy grave del servicio monitorizado.
- Unknown: Se da este estado cuando no es posible comprobar el estado de un servicio o los valores obtenidos no corresponden a ninguno de los estado anteriores.
- Pending: Servicio pendiente de ser comprobado.

En este apartado podemos ver si se encuentran activas las diferentes funcionalidades de Nagios y activarlas o desactivar según sea preciso. En la siguiente tabla aparecen las diferentes funcionalidades de nagios y su descripción.

Funcionalidad	Descripción
Flap detection	Detección de servicios o host que se encuentra fluctuando entre un estado y otro. Por ejemplo, si un servicio cada 5 minutos varia su estado de Critical a OK y viceversa; se marca con la etiqueta: Flapping service.
Notifications	Envío de notificaciones (Alertas) a clientes mediante los métodos predefinidos. Si se da el caso de que desactiváramos este servicio y la temperatura del simulador pasase a ser critica, no recibiremos ninguna alerta.
Event Handlers	Los Event Handlers (Manipuladores de eventos) son comandos que se ejecutan cuando una máquina pasa a un estado de Warning o critical, su función consiste en intentar de forma proactiva recuperar el servicio. Por ejemplo, en el caso de un molino aerogenerador, si la velocidad del viento es muy elevada habría que bloquear las aspas para evitar daños en la estructura. Dado este caso podríamos crear un Event Handler que bloque las aspas cuando la velocidad del aerogenerador sea superior a la recomendada.
Active Checks	Chequeos que ejecuta Nagios periódicamente para comprobar el estado de los servicios. Si se desactiva esta opción nagios deja de comprobar el estado de los servicios y host.
Pasive Checks	Se trata de los chequeos que se ejecutan de forma pasiva, es decir, cuando el servicio o el host detecta una anomalía envía el estado a Nagios. En otras palabras, Nagios recibe el estado sin consultarlo.

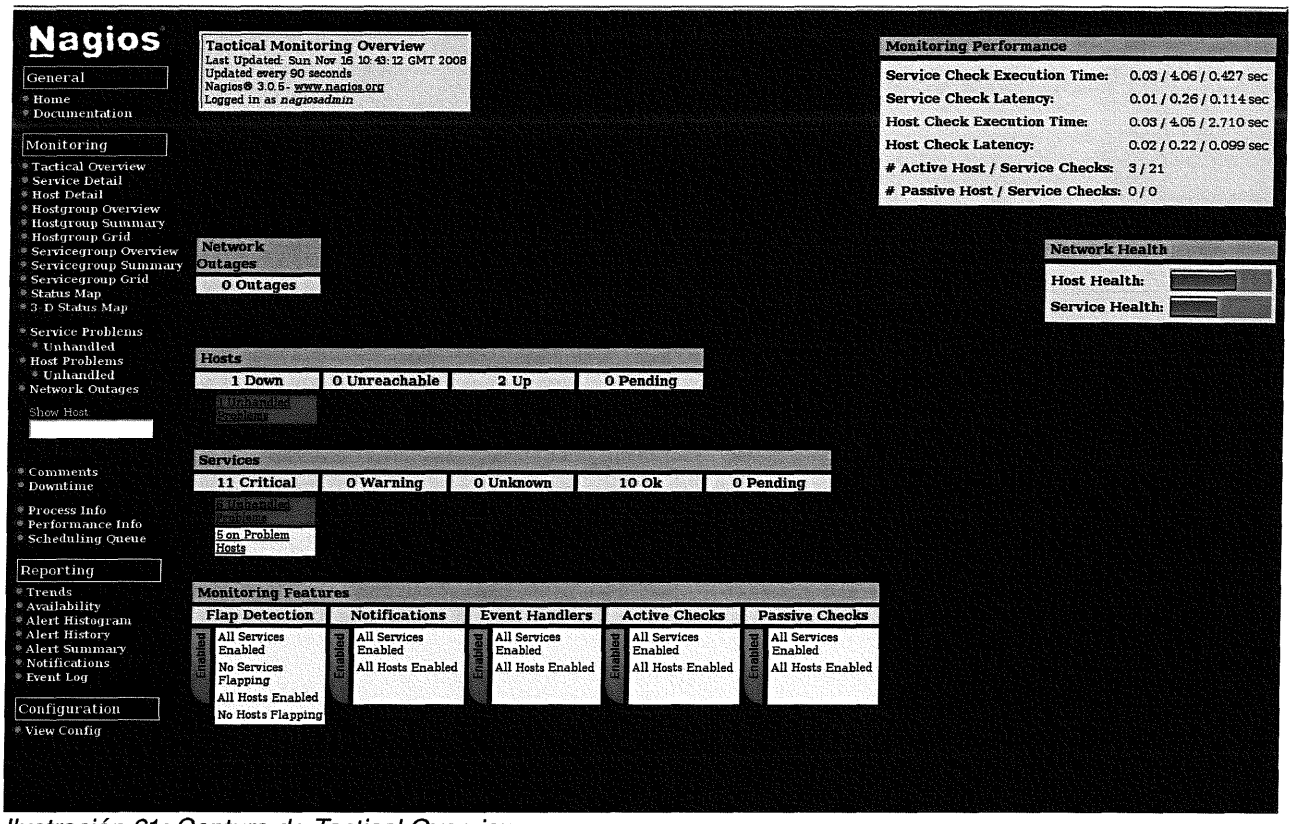

Ilustración 61: Captura de Tactical Overview

6.5.2 Service Detail

En este apartado de la interfaz de Nagios, aparece información detallada de todos los servicios dentro de una tabla. La información esta organizada por host y dentro de cada host aparecen los servicios asociados.

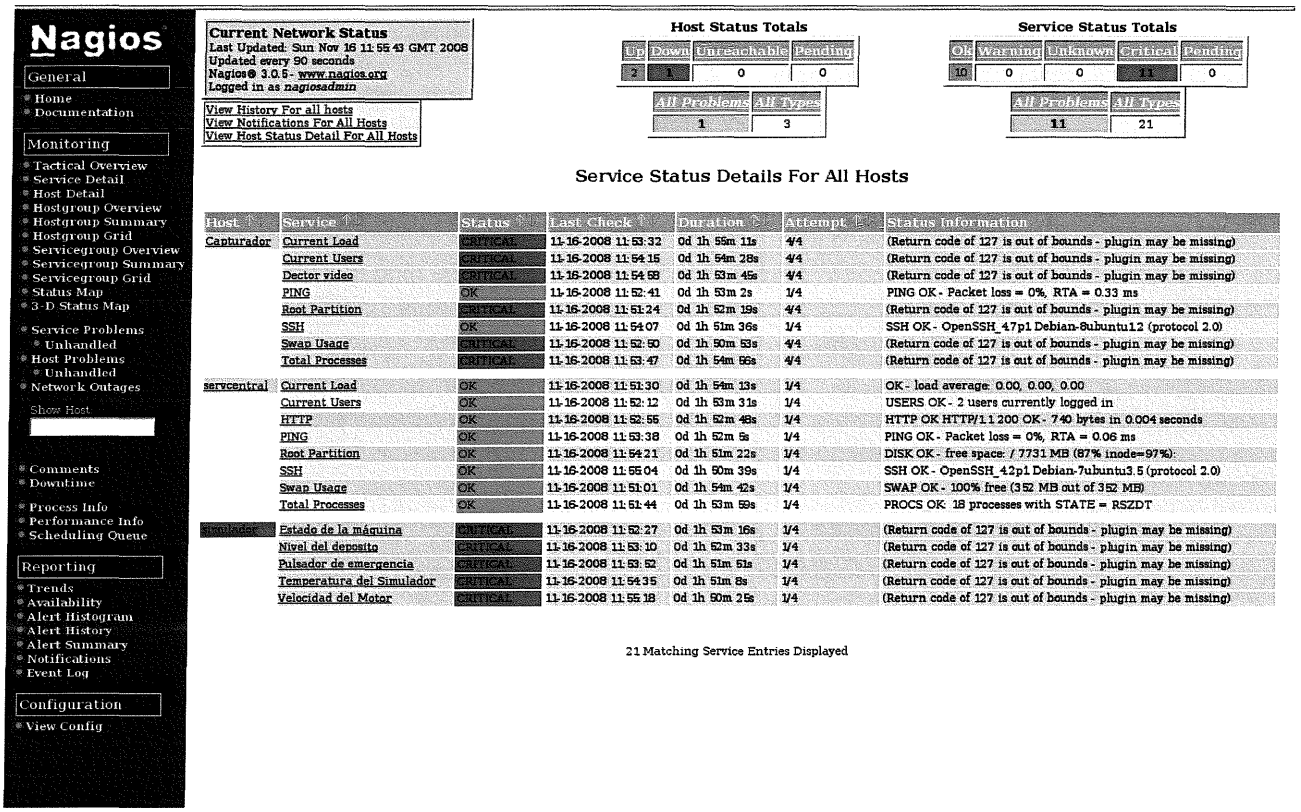

Ilustración 62: Captura del apartado Service Detail

A continuación se explican las diferentes columnas de la tabla del apartado Service Details, que aparecen en la Ilustración 62.

Columna	Descripción
Host	Mediante el color de fondo de la celda muestra el estado del host.
Service	Indica el nombre largo asociado al servicio. Cuelgan del host asociado.
Status	Estado en el que se encuentra el servicio desde la última actualización.
Last Check	Fecha de la última vez que se comprobó el estado del servicio.
Duration	Indica cuanto tiempo persiste el estado actual.
Attempt	Indica el número de comprobaciones que se han hecho antes de cambiar de estado
Status information	Muestra una pequeña descripción para ayudar al usuario a conocer la fuente del problema. En el caso de la temperatura el contenido de la columna podría ser: "Temperatura muy alta: 180ºC".

6.5.3 Otros apartados

En la siguiente tabla se describe la funcionalidad del resto de apartados de la interfaz web de Nagios.

Apartado	Descripción
Host detail	Muestra el estado de los diferentes host e información adicional.
Hostgroup overview Hostgroup Summary Hostgroup grid Servicegroup overview Servicegroup grid	Muestran información de los servicios de forma resumida, organizando la información en función de los grupos de host, es útil por ejemplo en caso de tener diferentes sedes para dividir los host por sedes.
Status map 3-D status map	Muestran el estado de los host sobre el esquema de red que las conecta.
Service problem	Como Service Detail pero solo muestra los servicios que no se encuentran en estado OK.
Host Problem	Como Host detail pero solo muestra los host que no se encuentran en el estado Up.
Comments	Permite a los usuarios dejar mensajes sobre los diferentes host y servicio. Se puede emplear por ejemplo para tener un historial de las actuaciones realizadas sobre los diferentes servicios o host.
Downtime	Se utiliza para desactivar las alertas sobre un host o servicio de forma programada. Por ejemplo si tenemos programada una actuación para la noche y no queremos recibir mensajes porque sabemos que la máquina estará inoperativa, podremos usar esta funcionalidad para desactivar las notificaciones.
Process Info	Información acerca del demonio del sistema Nagios. También nos permite ejecutar acciones sobre este como pueden paralo, reiniciarlo, etc.
Performance Info	Información y estadísticas sobre las tareas de chequeo.
Scheduling Queue	Cola de chequeos programados. Se puede desactivar el chequeo de un servicio.
Trends	Herramienta que permite crear gráficas donde se observa la evolución del estado de los servicio y host a lo largo del tiempo.
Availability	Herramienta que permite observar las estadísticas de disponibilidad de los servicios.
Alert Histogram	Permite la creación de gráficos para host, servicios, grupos de host o servicios, donde se observa la cantidad de alertas por tiempo emitidas.
Alert History Alert Sumary	Apartado donde aparecen todos los cambios de estados dados.
Notifications	Registro donde aparecen todas las notificaciones enviadas.
Event log	Registro de todos los movimientos registrados por Nagios, en el se incluyen cambios de estado, notificaciones, etc.

Además de los apartados vistos anteriormente también hay disponibles los apartados de cada host y servicio, donde hay información muy detallada de cada elemento. Además la interfaz web permite la programación eventos sobre los hosts y servicios.

7 Acceso desde red externa

Conectando la red de monitorización a internet se puede conseguir un acceso desde cualquier lugar a los servicios de monitorización y permite controlar desde un punto varias delegaciones o centros de producción. Cuando una red se conecta a internet hay que tener en cuenta que se expone a todo el mundo y es necesario por tanto, contar con medidas de seguridad para que únicamente puedan acceder a nuestro sistema los usuarios autorizados.

Las medidas de seguridad instaladas son dos, las redes VPN y el firewall.

7.1 VPN

Las redes VPN (Virtual Private Network) redes privadas virtuales son un sistema software que permite la interconexión entre máquinas y redes mediante conexión segura. Por tanto, se emplean para unir máquinas y redes de forma virtual, esto crea el efecto que todas las máquinas y redes dentro de la VPN se encuentran en la misma red local y se consige aprovechando los canales públicos de interconexión (Internet).

Las redes VPN se pueden crear entre routers, firewalls, PC, servidores, teléfonos móviles, etc. Una de las configuraciones típicas es crear una VPN entre los router de todas las sedes de una empresa, cada sede directamente conectada a todas las demás o en forma de estrella todas las sedes conectadas a la central. Esto permite compartir los servidores de correo, archivos, aplicaciones de administración, impresoras, etc; entre las diferentes sedes de forma segura. Por último, se pueden crear usuarios para que los que trabajen desde casa puedan conectarse a la red de la empresa.

En este proyecto se ha implementado una VPN mediante la instalación del servidor Openvpn en el servidor Nagios, lo que permite a los usuarios mediante un archivo de claves personal y el cliente Openvpn conectarse al sistema para acceder de forma segura a los servicios de la red de monitorización (Interfaz web de Nagios, visión del entorno por web, acceso a los servidores).

7.2 Firewall

Los Firewall o Cortafuegos, son sistemas de control de tráfico, analizan los paquetes[14] que pasan a través de ellos y según las reglas definidas, permiten el paso de los paquetes o no. Por tanto, se emplean para evitar intrusiones en sistemas, ataques y controlar el acceso a redes.

Pueden impedir el paso de los paquetes de dos formas: Rechazándolo (REJECT), de este modo el emisor recibe un paquete de respuesta conforme a que su paquete no ha sido aceptado. Ignorandolo (DROP), el paquete se bloquea pero no se responde, esto deja al emisor a la espera de una respuesta.

Los Firewall se instalan delante de la red que queremos proteger. Se identifica el trafico de salida como el que procede de la red protegida (Loc) al exterior (Net) y el tráfico de entrada es el procedente del exterior (Net) a la red protegida (Loc). En la Ilustración 63 el Firewall protege la red local (LAN) de la red de área amplia o externa (Wan).

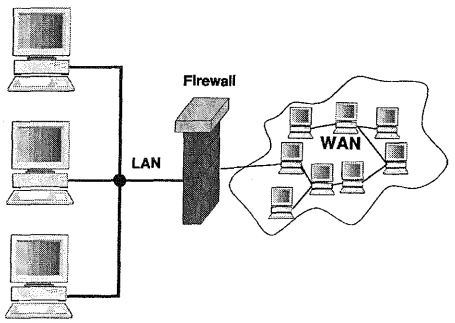

Ilustración 63: Localización Firewall

7.2.1 Iptables y shorewall

El Firewall en este proyecto se a implementado activado el modulo Iptables en el servidor Nagios que es el punto de entrada a la red de producción. Iptables es un modulo del núcleo de Linux que permite interceptar y manipular paquetes de la red.

La configuración de Iptables puede resultar lenta, complicada y difícil de entender a primera vista, ya que esta compuesta por un archivo donde se definen las reglas introduciendo líneas de configuración. Con la finalidad de evitar estos inconvenientes se ha instalado Iptables junto a Shorewall que es una herramienta diseñada para agilizar la configuración de Iptables, está aplicación, estructura la configuración en diversos ficheros, en los que se definen los diferentes objetos y funcionalidades (Interfaces, roles de las interfaces, reglas de tráfico, enmascaramiento, etc). Cuando se procede a la inicialización del Firewall Shorewall se encarga de traducir su configuración a reglas de Iptables.

14 En referencia a los paquetes TCP/IP.

8 Servidor externo de control de conectividad

El servidor externo de control de conectividad es una máquina que se encarga de comprobar que el sistema de monitorización es accesible desde internet. Cuando el sistema deja de ser accesible, alerta al usuario para que actué en consecuencia.

Este sistema es útil en casos en los que la única vía de comunicación con el usuario es a través de internet. Por ejemplo, si disponemos de un servicios de alertas por SMS empleando la red de telefonía móvil y mediante correo electrónico, no sería necesario el sistema de control de conectividad porque el propio sistema de monitorización emplearía la conexión GSM para enviar la alerta. En cambio, si para enviar las alertas SMS empleamos un proveedor de internet; en caso de perdida de conexión de la red, no recibiríamos ninguna alerta. Lo cual puede ser un grave problema si salta otra alarma en el sistema.

Se puede implementar de diversas formas el servicio externo de aviso de conectividad:

- Se puede implementar el sistema empleando un servidor de monitorización (Nagios) externo a la empresa.
- Se puede contratar un servicio con el proveedor de la conexión para que en caso de caída nos alerte.
- Otra opción consiste en emplear un servidor externo que ejecute una aplicación que realice un ping continuo y en caso de no recibir respuesta envíe un correo electrónico de aviso.

Esta parte del proyecto no se ha integrado en la maqueta porque supone emplear una máquina adicional y el sistema ya dispone de un segundo canal de comunicación con el usuario, el canal GSM.

9 Valoración económica del proyecto

Los costes de este proyecto se han dividido en dos grupos, costes iniciales, en otras palabras los costes que supondrá el proyecto desde su inicio hasta la entrega y costes de mantenimiento, una vez entregado el producto cuanto costará mantenerlo cada año.

Para analizar los costes de este proyecto se ha tenido en cuenta el precio de la hora en función del tipo de actividad realizada, el número de horas empleadas y el precio de los materiales empleados.

9.1 Costes en función del tipo de actividad

En la siguiente tabla aparecen los costes de la hora para cada tipo de actividad. El coste/hora se calcula a partir de la relación entre tipo de actividad y salario del perfil profesional que la realiza. Por ejemplo, perfil un analista, perfil programador, etc.

Actividad	Coste/Hora
Análisis	80 €/Hora
Diseño	80 €/Hora
Programación	60 €/Hora
Investigación	70 €/Hora
Documentación	60 €/Hora
Pruebas	60 €/Hora
Aprovisionamiento	60 €/Hora

9.2 Coste humano del proyecto

Teniendo en cuenta los datos del apartado anterior y la planificación del proyecto, se ha elaborado la siguiente tabla donde aparece para cada tarea la descripción, el número de horas empleado, el coste/hora asignado y el precio total. En la última fila aparece la suma total de horas que dura el proyecto y el coste total.

Descripción tarea	Horas invertidas	Coste/Hora	Coste total
Viabilidad del proyecto	14 horas	80 €	1120 €
Diseño de la estructura del sistema	14 horas	80 €	1120 €
Aprovisionamiento de materiales	10 horas	60 €	600 €
Estudio de la placa de demostración	20 horas	70 €	1400 €
Implementación del software de la placa	40 horas	60 €	2400 €
Investigación comunicación placa/terminal	6 horas	70 €	420 €
Implementación software comunicación placa/terminal	14 horas	60 €	840 €
Estudio servicio de monitorización	15 horas	70 €	1050 €
Diseño servicio de monitorización	15 horas	80 €	1200 €
Implementación sistema de monitorización	30 horas	60 €	1800 €
Estudio servicio de correo	5 horas	70 €	350 €
Implementación del servicio de correo	5 horas	60 €	300 €
Estudio envío de SMS	10 horas	70 €	700 €
Implementación envío de SMS	10 horas	60 €	600 €
Estudio VPN	6 horas	70 €	420 €
Diseño VPN	2 horas	80 €	160 €
Implementación VPN	6 horas	60 €	360 €
Estudio servidor externo de conectividad	4 horas	70 €	280 €
Estudio sistema de vídeo	12 horas	70 €	840 €
Implementación del sistema de vídeo	8 horas	60 €	480 €
Simulación y corrección de errores	40 horas	60 €	2400 €
Documentación	99 horas	60 €	5940 €
Total:	***385 horas***		***24780 €***

9.3 Coste de los materiales

En la siguiente tabla se especifica el coste que supondrá cada elemento empleado en el proyecto.

Elemento	Coste
Laboratorio PIC School	160 €
PC capturador	400 €
Webcam	20 €
PC Servidor	400 €
Teléfono móvil	75 €
Switch	20 €
Cableado de red	15 €
PC externo	400 €
Total	***1490 €***

9.4 Coste inicial del proyecto

Se trata del coste total que supondrá la implementación de este proyecto, momento en que se entrega al cliente. Es la suma del coste humano más el coste total de los materiales. El coste total es de **34732 €**.

9.5 Mantenimiento anual del sistema

En la siguiente tabla se calcula el coste que supondrá mantener el sistema, a partir del desglose de las tareas.

Descripción tarea	Cantidad	Precio unidad	Coste total
Gestión /mantenimiento de los equipos	36 horas	20 €	720 €
Alertas SMS	12 facturas	11 €	132 €
Total			***752 €***

El calculo del precio de la gestión y mantenimiento de los equipos se ha basado en el empleo de 3 horas mensuales de gestión de los equipos. El coste total de las alertas se han calculado teniendo en cuanta que cada mes se producen 30 alertas notificables por SMS y la tarifa mínima de contrato para la línea de teléfono móvil es de 11 €.

10 Conclusiones

10.1 Objetivos cumplidos y resultados

Teniendo en cuenta los objetivos marcados al inicio del proyecto a continuación se exponen las conclusiones obtenidas en base a los objetivos marcados y los resultados obtenidos.

Si bien es cierto que principal se ha diseñado un sistema de monitorización de una placa de demostración, también es cierto que se han incluido todos los elementos involucrados en el proceso de monitorización de una máquina de produción. Cada elemento se puede ampliar y adaptar a las necesidades del cliente al que se aplican. Por ejemplo, si un cliente dispone de 20 máquinas de producción, primero se introducen nuevos sensores o se adaptan los que ya disponen las máquinas para que mediante varios PC capturadores de sensores se puedan recoger los valores que son interesantes para la monitorización. Por tanto, dada la adaptabilidad comentada queda justificado que se trata de un sistema universal.

Todos los procesos se encuentran controlados desde el ordenador central, y se puede acceder a toda la información a través de este, por tanto se cumple que el sistema está centralizado.

Dado un número elevado de máquinas se requiere un gran número de técnicos que las revisen, las reparen y recarguen cuando es necesario. Si introducimos el sistema de monitorización aquí diseñado, el número de técnicos que realiza esta labor, la periodicidad con la que se ha de revisar las máquinas, la situación en que una máquina se para porque no dispone de suministros, no ha sido recargada a tiempo. Se reduce porque en todo momento se conoce el estado de las máquinas sin tener que pasarse a visitarlas. Por tanto, reducimos costes con lo que la implementación del proyecto es cada vez más rentable cuanto mayor es el número de máquinas y mayor es el tiempo transcurrido desde la implementación del sistema. Por tanto, teniendo en cuenta estas premisas se puede asegurar de que se trata de un sistema de bajo precio y rentable.

El mantenimiento del sistema consiste en reparar los equipos averiados, ajustar los parámetros, los costes en cuanto a envíos de SMS. En relación a las ventajas ofertadas el coste del sistema, una vez implementado, es muy bajo.

Se a conseguido mantener informado al usuario en todo momento de los problemas incluso en los casos en que la conexión a internet del sistema ha estado caída gracias al sistema de avisos por SMS, por tanto el sistema informa al usuario en todo momento del estado del sistema sin que este tenga que estar pendiente y de forma activa preocupándose por el sistema

de producción. Por así decirlo puede dormir relativamente tranquilo porque si algo falla sonará su teléfono móvil.

Mediante la aplicación de un firewall que restringe el acceso al sistema y permite solo acceder algunos puertos, el requerimiento de contraseñas, usuarios para acceder a los sistema y la implementación de una VPN garantizan el acceso restringido, de forma segura al sistema y la protección de los datos.

Mediante la posibilidad de colocación de cámaras al sistema de forma económica se puede visualizar el entorno y detectar movimientos.

El tiempo transcurrido entre que un sensor presenta valores anómalos y el usuario recibe la alerta no supera los 5 minutos, por tanto la monitorización es casi instantánea.

10.2 Planificación resultante

En la realización del proyecto se ha seguido el orden de los pasos según lo planificado porque cada elemento nuevo necesitaba del anterior para poder funcionar correctamente. El proyecto se inició con un estudio de viabilidad, en el que se buscó información sobre sistemas similares, herramientas disponibles en el mercado para implementar el sistema, se realizó una planificación para comprobar que el tiempo necesario para implementar el sistema no superaba un cuatrimestre de trabajo teniendo en cuenta que se dedicarían 2 horas y media diarias. Tras este estudio la conclusión fue, el proyecto es viable y entonces se empezó a trabajar en el diseño.

Conociendo, gracias al paso anterior, como están montados los sistemas de monitorización y teniendo en mente antes del proyecto la forma del sistema, el diseño definitivo del proyecto llevó menos tiempo del planificado.

Con el diseño como guía se inició la búsqueda de una placa de simulación que contuviera los elementos suficientes para simular una máquina de monitorización y ofreciese la posibilidad de comunicarse con un PC. Se escogió la placa PIC'SCHOLL, el procesador a instalar PIC16F876A y el compilador SDCC; se estudiaron. Conociendo los elementos, se diseñó el programa de control de la máquina implementando todas las funciones planificadas, captura de sensores y intercambio de información con el capturador de sensores. Además se ha empleado la pantalla LCD para ver en el momento el valor de los sensores, lo que ha facilitado la depuración del programa. La implementación del programa, ha llevado más tiempo del planificado porque durante el desarrollo de este surgieron errores, que no se pudieron solucionar rápidamente, el retraso ha sido motivado por el hecho de que no había suficiente documentación y ejemplos sobre el compidador SDCC.

Una vez la placa se encontraba preparada para enviar el valor de sus sensores se inició el proceso de configuración del PC capturador de sensores y el diseño del programa de captura de los valores de la placa. Fue necesario hacer cambios en el programa de la placa para conseguir un protocolo de comunicación apropiado. La instalación e inclusión de la cámara web en el sistema fue la tarea que menos porcentaje de tiempo requirió en proporción al tiempo estimado, se planificaron 25 horas y finalmente se pudo completar en 10 horas.

En la instalación del servidor central se han empleado menos horas de las planificadas, principalmente porque el sistema de monitorización ya era conocido, no obstante se profundizaron los conocimientos en la parte de configuración y se descubrieron mucha posibilidades de las que ofrecen. La parte del correo ha llevado más tiempo del esperado porque los servidores de internet disponen de políticas muy restrictivas de Spam y al intentar enviarles correos nunca llegaban porque eran bloqueados. Finalmente se decidió que el correo se entregaría en local y además se decidió que sería más conveniente porque la maqueta no dependería de la conexión a internet para funcionar.

Se planificó el empleo de un servidor de control externo de conectividad, sin embargo finalmente no se ha implementado porque en caso de fallo de conectividad se informa al cliente vía GSM.

La última fase ha consistido en la simulación y corrección de errores, no ha sido necesario modificar ningún aspecto de la programación realizada ya que el sistema funcionaba correctamente, sin embargo si que se han retocado los parámetros de Nagios porque las alertas tardaban más tiempo del esperado en llegar.

En conjunto el tiempo empleado al proyecto ha sido muy similar al estimado, sin embargo la fecha de finalización no ha sido la misma debido a que no se ha cumplido la dedicación de horas diarias.

10.3 Mejoras y futuro trabajo

Una vez concluido el proyecto se han ampliado los conocimientos sobre el campo estudiado, esto permite detectar vulnerabilidades, presentar mejoras y nuevas funcionalidades, que no se tuvieron en cuenta en el momento de la realización del proyecto. Asimismo se aprovecha este apartado para proponer más funcionalidades que se habrían implementado si se hubiera invertido más tiempo o que se podrían desarrollar en futuros proyectos.

10.3.1 Sistema de aviso inmediato

La primera propuesta para mejorar el sistemas es el permitir que anomalías en determinados servicios críticos se puedan notificar tan pronto como se producen, esto se puede conseguir mediante el empleo de chequeos pasivos en Nagios. El tiempo de notificación al usuario máximo actual es de 5 minutos, pero existen servicios que por su naturaleza e urgencia es importante que se informe cuanto antes al usuario para que pueda actuar. Un ejemplo muy crítico sería un fallo en el reactor de una central nuclear, en este caso cuanto antes se notifique el problema más fácil será paliarlo.

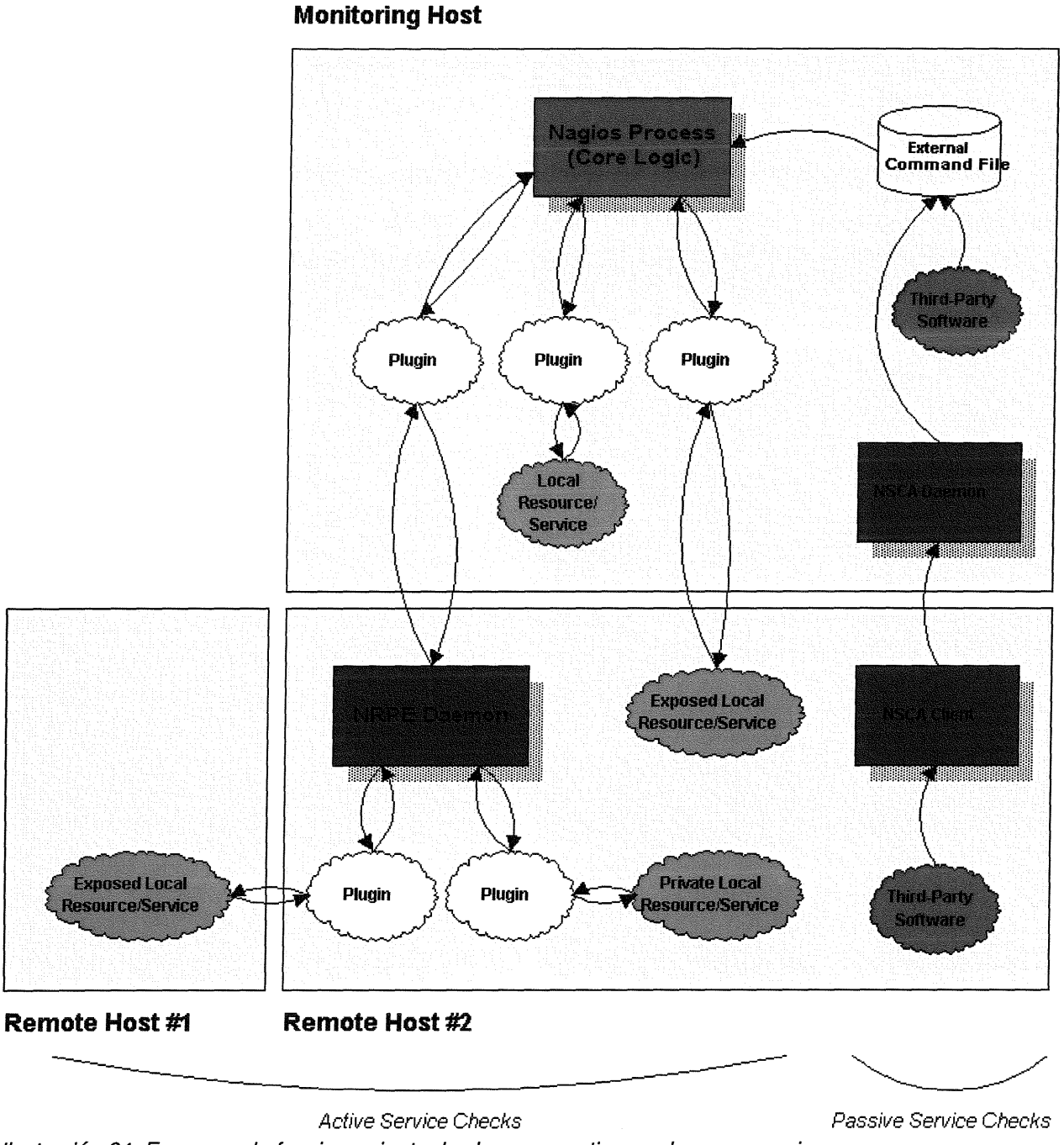

Ilustración 64: Esquema de funcionamiento de chequeos activos y chequeos pasivos

Para implementar esta mejora el PC capturador, a diferencia de como funciona ahora, que es él el que espera a que el servidor central le consulte el estado de los sensores, tendría que ser proactivo y enviar la alerta al servidor central; Por último el servidor central informaría al usuario inmediatamente.

10.3.2 Base de datos para el control de la producción

El sistema de monitorización maneja un gran número de información sobre las máquinas que monitoriza y de estos datos se pueden extraer informaciones como el tiempo que están trabajando, tiempo que están averiadas, consumo de un cierto material. Incluso se podría agregar sensores que se dediquen exclusivamente a recoger datos como contar el número de piezas producidas, contar el tiempo que tarda cada pieza en recorrer desde el inicio de la cadena al final, etc.

Los datos se pueden almacenar en una base de datos y con el debido software analizarlos para mejorar los procesos de producción. Esta línea de trabajo es muy interesante porque puede aportar muchos beneficios y no requiere un coste muy superior una vez implementado el sistema de monitorización.

11 Bibliografía

Toda la información necesaria para la elaboración de este proyecto se ha extraído de internet y los manuales del software instalado. A continuación se detallan los contenidos web y herramientas más utilizados en este proyecto.

11.1 Fuentes de información

El buscador Google (http://www.google.com)

La página oficial de Nagios (http://www.nagios.org)

La web de la empresa Microchip (http://www.microchip.com)

La página oficial del compilador SDCC (http://sdcc.sourceforge.net)

Los foros de microcontroladores PIC (http://www.todopic.com.ar/foros)

La documentación de la web de shorewall (http://www.shorewall.net)

La web de bulma dedicada a linux y software libre (http://www.bulma.net)

La web ubuntu-es dedicada a la distribución ubuntu (http://www.ubuntu-es.org/)

El foro oficial de Nokia (http://discussion.forum.nokia.com/forum)

La web de Microsystems engineering (http://www.msebilbao.com)

Proyecto final de carrera de Francisco Javier Pérez Gulias: *"Monitorizació de un Sistema MicroComputador".*

Apuntes de las asignaturas: Sistemas digitales y microcontroladores (SDMI) y Periféricos e interficies (PI).

11.2 Herramientas empleadas

A continuación aparece el listado de herramientas y software empleadas en el proyecto más destacables.

- Openoffice, software de oficina. Documentación y esquemas.
- Dia, esquemas de red.
- Nagios, sistema de monitorización.
- Ubuntu, sistema operativo.
- PIC'SCHOLL placa de desarrollo y aprendizaje para PIC.
- Nokia N70, teléfono móvil.
- SDCC, compilador de PIC.
- GCC, compilador de C para linux.
- Portatil Dell XPS M1530, PC de capturador de imágenes.
- PC sobremesa clónico Pentium III, servidor central.
- PIC16F876A microcontrolador.
- Postfix, agente de transporte de correo (MTA).

12 Anexo A: Definiciones

Diagrama de flujo: Un diagrama de flujo es un esquema que se emplea para representar el flujo de ejecución del programa, describe el funcionamiento del programa visualmente.

Flanco: En una señal digital, se denomina flanco a la transición del estado de la señal. Flanco de subida a la transición de 0 a 1 (En rojo en la imagen) y flanco de bajada a la transición de 1 a 0 (En azul en la imagen).

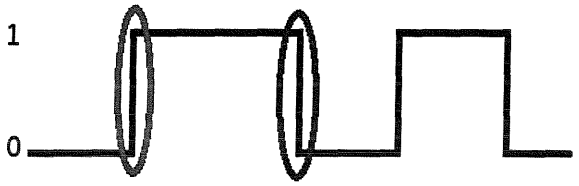

Ilustración 65: Flancos

GSM: El Sistema Global para las Comunicaciones Móviles (GSM) es un sistema estándar para comunicación, muy utilizando teléfonos móviles. Permite enviar y recibir llamadas, mensajes por e-mail, faxes, navegar por Internet, así como utilizar otras funciones digitales de transmisión de datos, incluyendo el Servicio de Mensajes Cortos (SMS).

Interfaz: Es el puerto (Circuito físico) que permite a dos sistemas o dispositivos comunicarse entre si. Existen diversos estándares (Interfaces RS-232, RJ-45, USB,...).

http://es.wikipedia.org/wiki/Interfaz

Jumper: Se trata de un elemento para interconectar dos terminales de manera temporal sin tener que efectuar una operación que requiera herramienta adicional, dicha unión de terminales cierran el circuito eléctrico del que forma parte.

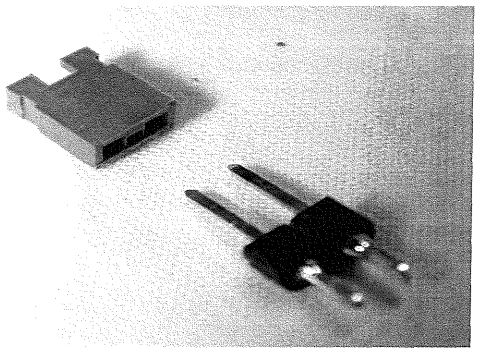
Ilustración 66: Jumper y punto de conexión

Microcontrolador: Un microcontrolador es un circuito integrado o chip que incluye en su interior las tres unidades funcionales de una computadora: CPU, memoria, unidades de Entrada/Salida, es decir, se trata de un computador completo en un solo circuito integrado.

http://es.wikipedia.org/wiki/Microcontrolador

Monitorización: Consiste en conocer cual es el estado de los equipos de una instalación y generar alertas para tomar las decisiones oportunas, obteniendo en cada momento la mayor información posible de forma automática.

Sistema operativo: Un sistema operativo es un software que ofrece una interfaz a usuario para hacer más amigable el uso de una máquina (PC, móvil, PDA, etc). Permite la ejecución de programas, el uso y gestión del hardware. Ejemplos de sistemas operativos son: Linux, Unix, Windows, Mac OS.

SMS: Servicio de mensajes cortos, es un servicio que permite el envío de mensajes de texto corto entre teléfonos móviles, teléfonos fijos y otros dispositivos. SMS fue diseñado originariamente como parte del estándar de telefonía móvil digital GSM, pero en la actualidad está disponible en una amplia variedad de redes, incluyendo las redes 3G.

PDIP (Plastic-Dual-Inline-Package): Tipo encapsulación de dispositivos electrónicos en forma de rectángulo con dos filas paralelas de pestañas.

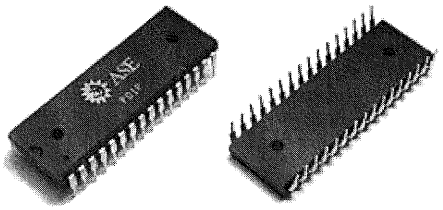

Ilustración 67: Componentes encapsulados según PDIP

Protoboard o placa de pruebas: Es una placa de uso genérico reutilizable, usada para construir prototipos de circuitos electrónicos sin soldadura. Contiene diversos agujeros conductivos interconectados por filas, que se pueden unir entre sí empleando cable o componentes electrónicos, para formar circuitos.

Paquetes TCP/IP: Se trata de agrupaciones de bit lógicas que contienen una cabecera IP, seguida de una cabezera TCP y por último contienen la información que intercambian las máquinas (Por ejemplo, el valor de los sensores, una parte de una web, etc). El objetivo de la cabecera IP es hacer llegar de una máquina a otra un paquete de información, el protocolo IP no se preocupa de si el paquete llega al destino. Por el contrario el protocolo TCP asegura si el paquete llega al otro extremo y caso contrario aplica medidas para corregir errores como el reenvío de paquetes.

UART (Transmisor y Receptor Asíncrono Universal): Estandar de comunicación para equipos que soporta 115.2 Kbps, la comunicación es asíncrona.

USART (Transmisor y Receptor Sincrono asíncrono Universal): La diferencia con el UART es que en este caso la transmisión es sincronizada.